UNIVERSITY OF SÃO PAULO
PHYSICS INSTITUTE

Natalí Soler Matubaro de Santi

Machine Learning methods for extracting cosmological information

São Paulo

2024

Universidade de São Paulo
Instituto de Física

Métodos de aprendizado de máquina para obtenção de informações cosmológicas

Natalí Soler Matubaro de Santi

Orientador: Prof. Dr. Luis Raul Weber Abramo

Tese de doutorado apresentada ao Instituto de Física da Universidade de São Paulo, como requisito parcial para a obtenção do título de Doutora em Ciências.

Banca Examinadora:
Prof. Dr. Luis Raul Weber Abramo - Orientador (IF-USP)
Prof. Dr. Laerte Sodré Junior (IAG-USP)
Prof. Dr. Clécio Roque de Bom (CBPF)
Prof. Dr. Ravi Kiran Sheth (UPENN)
Prof. Dr. Raúl Esteban Angulo de la Fuente (DIPC)

São Paulo
2024

FICHA CATALOGRÁFICA
Preparada pelo Serviço de Biblioteca e Informação
do Instituto de Física da Universidade de São Paulo

Santi, Natali Soler Matubaro de

Métodos de aprendizado de máquina para obtenção de informações cosmológicas / Machine learning methods for extracting cosmological information. São Paulo, 2024.

Tese (Doutorado) - Universidade de São Paulo. Instituto de Física. Depto. de Física Matemática.

Orientador: Prof. Dr. Luis Raul Weber Abramo.
Área de Concentração: Física.

Unitermos: 1. Cosmologia; 2. Astrofísica, 3. Inteligência Artificial.

USP/IF/SBI-047/2024

University of São Paulo
Physics Institute

Machine learning methods for extracting cosmological information

Natalí Soler Matubaro de Santi

Supervisor: Prof. Dr. Luis Raul Weber Abramo

Thesis submitted to the Physics Institute of the University of São Paulo in partial fulfillment of the requirements for the degree of Doctor of Science.

Examining Committee:
Prof. Dr. Luis Raul Weber Abramo - Supervisor (IF-USP)
Prof. Dr. Laerte Sodré Junior (IAG-USP)
Prof. Dr. Clécio Roque de Bom (CBPF)
Prof. Dr. Ravi Kiran Sheth (UPENN)
Prof. Dr. Raúl Esteban Angulo de la Fuente (DIPC)

São Paulo
2024

Aos meus pais, Aparecida e Ricardo.

As opiniões, hipóteses e conclusões ou recomendações expressas neste material são de responsabilidade da autora e não necessariamente refletem a visão da FAPESP.

The opinions, hypotheses, and conclusions or recommendations expressed in this material are the author's responsibility and do not necessarily reflect the views of FAPESP.

ACKNOWLEDGEMENTS

Essa tese é dedicada a todos que contribuíram para que eu conseguisse chegar até aqui!

Primeiramente, gostaria de agradecer ao Raul Abramo pela orientação e por todo o tempo dedicado a minha pessoa. Muito obrigada por ter aceitado me orientar, pela confiança no meu trabalho (mesmos tendo vindo de uma área muito diferente), pelo apoio (tanto profissional quanto pessoal), pela amizade, entusiasmo e motivação constantes, por toda a paciência (para todas as perguntas pelas quais eu já deveria saber a resposta e para quando eu travava) e vontade de trabalharmos juntos. Agradeço por termos trabalhado em temas desafiadores, por todas as oportunidades proporcionadas e pela confiança a partir do momento que passei a ser mais independente na pesquisa. Não posso deixar de agradecer a você e a Elena por terem topado ser meus padrinhos de casamento.

En segundo lugar, agradezco a Francisco Villaescusa-Navarro por la co-orientación y por la oportunidad de poder salir de Brasil por primera vez en la vida, permitiéndome pasar un año en Nueva York. Gracias por haber confiado en lo que venía desarrollando con Raul, por haberme permitido trabajar en uno de los mejores centros de Cosmología y aprendizaje de máquina del mundo, por toda su voluntad de enseñarme y ayudarme, por todos los contactos que pude construir durante este período y por todo el apoyo profesional y personal (en uno de los mayores desafíos de mi vida). Creo que nunca podré retribuir todas las oportunidades que se me han brindado, de poder haber presentado los resultados de Este a Oeste de los Estados Unidos (y Canadá) y por el préstamo de su Specialized (mi sueño de bicicleta, muy cara en Brasil, y aún XG – un desafío aparte para la bajita aquí), para andar en bicicleta en NYC durante todo un año.

Aos meus pais, Aparecida e Ricardo, agradeço por todo o amor e confiança depositados em mim. Obrigada por me apoiarem e me aturarem em todos os momentos da minha vida. Foram vocês que me derrubaram e, então, ensinaram a me levantar, me transmitiram muito mais que valores, vocês moldaram a minha personalidade e, acima de tudo, me ensinaram a nunca desistir dos meus sonhos. Espero ter conseguido seguir alguns dos passos que vocês tanto almejaram, mas que a vida e suas eventualidades não permitiram. Saibam que todos os sacrifícios de vocês são muito valorizados. Vocês são e sempre serão meus melhores exemplos e inspiração. Sem vocês eu não seria nada e eu só cheguei até aqui por causa de vocês!

Agradeço aos meus irmãos, Nicole e Mario, que, além de me apoiarem, sempre me ajudaram a deixar o trabalho um pouco de lado e a rir das situações mais inusitadas possíveis. Nossa interação vai muito além de compartilharmos o dia a dia, segredos, felicidades e frustrações, inclui algo que só ocorre entre irmãos, a naturalidade da afinidade. Peço desculpas por não estar presente (oficialmente, mas estou em pensamento) em todos os momentos de suas

vidas, mas tenham certeza que me orgulho muito e sempre me orgulharei de quem vocês se tornaram, dois futuros e primeiros engenheiros da família. Afinal, meus melhores amigos!

Ao meu marido Jasiel, agradeço por termos encontrado juntos uma das peças fundamentais da vida chamada amor. Ah, como a vida é boa ao seu lado. Agradeço por você me acompanhar, me amar desse jeito louco e por me apoiar em cada novo passo, diariamente (no presente e no futuro). Obrigada por ser meu mestre motociclista (calma que eu ainda te alcanço), por tudo o que temos vivido e pelo que sonhamos em desbravar juntos. Estamos aos pouquinhos construindo a nossa família. Ti amo!

Quero agradecer a todos os meus familiares, minhas tias Angela, Delma, Maria, Dirce e Eliane, meus tios Adilson, Paulo, Carlos, Zé, Neco e Jorge, meus primos Peterson, Emerson e Jonathan, minhas primas Jéssica, Jennifer, Suelen, Rafaela e Tata, e minha avó Bá que completam a minha existência. Obrigada pelas conversas, pela amizade e por sempre me receberem de braços abertos não importando a hora ou a situação.

Gostaria de agradecer a todos os professores que contribuíram para a minha formação, em especial destaco aqui aqueles que tive o prazer de ter um maior contato. À minha tia e professora Angela, que sempre me incentivou a seguir no ambiente acadêmico. À minha professora da terceira série Cláudia Maria Botassi, para a qual eu prometi ser astronauta, mas saí como física. Aos meus orientadores e co-orientadores Antonio Carlos Hernandes, Ariane Baffa Lourenço, Marcelo Rubens Barsi Andreeta, Thiago Martins Amaral, Attilio Cucchieri, Raphael Santarelli, Antonio Montero-Dorta e, novamente, Raul Abramo e Francisco Villaescusa-Navarro. Aos meus professores de graduação e pós-graduação Rodrigo Gonçalves Pereira, Luis Nunes de Oliveira, Leonardo Paulo Maia, José Fabián Schneider, Tereza Mendes, Máximo Siu Li, Paulo Daniel Emmel, Marcos Lima, Laerte Sodre Júnior, Claudia Mendes de Oliveira, Clécio Roque de Bom, Alice Pisani, David Spergel e Elisa Ferreira. Obrigada por tudo o que vocês me ensinaram tanto acadêmica quanto pessoalmente. Não posso deixar de agradecer a Ana Laura Batista, por todo o acompanhamento durante meu doutorado.

Às minhas amigas desde o colégio Carol, Julia e Mayara. Aos meus amigos de mundo acadêmico e não-acadêmico Guilherme M. L., Peterlevitz, Lício, Noel, Alfredo, Fábio, Viviane e Lívia. Aos meus comparsas de Cosmologia e Astronomia do Brasil: Natália Rodrigues, Antonio Montero-Dorta, Tiago Castro e Pablo Araya Araya (os quatro, por toda a amizade e colaborações), Vitor Cernic, Erik Vinívius, Lilianne Nakazono, Thiago Mergulhão, Daniel e Rafael Gomes, Carolina Queiroz, Francisco Germano Maion, Caroline Guandalin, Rodrigo Voivodic, Renan Boschetti, Beatriz Tucci, João Ferri, Ian Tashiro e Lukas Liland. To the friends from New York, Lucia Perez, Helen Shao, Bonny Wang (the three, for all the friendship and collaborations), Elena Hernández-Martínez, Giulio Fabian, Adriaan Duivenvoorden, Adrian Bayer, Fedir Boriko, and Oriol Castander. And to my friends from Paco's group, Arnab Lahiry, Chaitanya Chawak, Nicolás Echeverri Rojas, and Pablo Villanueva-Domingo: we will still meet in person around the world! Muito obrigada pela amizade, conversas e por tudo o que pude aprender (e alcançar)

na companhia (e com a ajuda) de vocês.

Não posso esquecer de agradecer aos meus companheiros Tesla e Dino. Tesla, que passou quase todas as horas de pandemia literalmente ao meu lado (seja no chão ou na minha cama), que quase levanta voo abanando o rabo de felicidade ao me ver e que me leva para passear para que eu me sinta viva. Você se tornou meu irmão e Titio do Dino. Dino, esse filhotão do qual eu virei mamãe, que tem me ensinado como viver feliz e em festa todos os dias, conseguindo superar a loucura do menino Tesla. Agora é você que me leva para passear.

Por fim, agradeço ao Instituto de Física (IF), à Universidade de São Paulo (USP), ao Instituto Flatiron, à Fundação Simons, ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo auxílio direto e/ou indireto para a possibilidade de realização do meu doutorado. Atenção devida a todos os funcionários, de ambos os locais, que me auxiliaram em todas as etapas do doutoramento, seja pela amizade e/ou manutenção do local no dia-a-dia, seja por me guiar em processos burocráticos. Mais especificamente, agradeço à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processos números 2019/13108-0 e 2022/03589-4, pelo apoio financeiro para o mesmo. And to the Ph.D. committee for their availability to evaluate this work.

"Science, my lady, has been built upon many errors;
but they are errors which it was good to fall into, for they led to the truth."
Journey to the centre of the Earth - Jules Verne

"Lutarei enquanto existir amor, até o fim da minha vida."
Estrela fascinance patrine

ABSTRACT

The new era of cosmological observations is generating vast amounts of data, increasing the pressure for improvements in both existing and new techniques to analyze this data. Machine Learning (ML) methods are currently at the cutting edge in terms of new techniques and tools, often surpassing traditional methods. In this work, we employ a series of ML techniques to: (1) improve cosmological covariance matrices, (2) investigate the halo-galaxy connection, and (3) perform robust field-level likelihood-free inference with galaxies and halos. Parameter inference is a key aspect in Cosmology, and here we present two different approaches: the use of traditional methods, aimed at obtaining accurate and precise cosmological covariance matrices using image denoising techniques, and a novel approach, which involves deriving parameters directly by converting galaxy/halo catalogs into graphs, without cuts on scale, and then feeding these graphs into graph neural networks to predict the parameters. Simultaneously, the relationship between galaxies and halos is central to describing galaxy formation and is a fundamental step towards extracting precise cosmological information from galaxy maps. We address this problem with a sequence of approaches, ranging from using raw methods and augmenting the data set to stacking methods and converting a regression problem into a classification one, to recover galaxy properties along with their stochasticity. All of these projects aim at improving the extraction of information from simulations by enhancing the accuracy and precision of the derived constraints, thereby impacting cosmological parameters and the halo-galaxy connection. These are the initial steps before applying this new set of innovative methodologies to real data, for both current and next-generation surveys.

Keywords: Cosmological parameters inference. Machine learning algorithms. Cosmological simulations. Halo-galaxy connection.

RESUMO

A nova geração de observações cosmológicas está gerando uma enorme quantidade de dados e aumentando a pressão pelo desenvolvimento de novas e já existentes técnicas para a análise dos mesmos. Métodos de Aprendizado de Máquina (AM) tem se mostrado como uma excelente e revolucionária alternativa para essa função, muitas vezes superando os métodos tradicionais. Neste trabalho nós utilizamos uma série de técnicas de AM para: (1) melhorar matrizes de covariância cosmológicas, (2) investigar a conexão galáxia-halo, e (3) realizar uma inferência robusta de parâmetros cosmológicos, livre de verossimilhança, usando apenas o campo de galáxias e halos. A inferência de parâmetros é uma atividade central em Cosmologia e aqui nós apresentamos duas diferentes soluções: o uso de métodos tradicionais, para a obtenção de matrizes de covariância cosmológicas precisas e acuradas por meio uma técnica de remoção de ruído de imagens e um método novo, que envolve converter catálogos de galáxias e halos em grafos para alimentar uma rede neural gráfica capaz de diretamente inferir os parâmetros. Ao mesmo tempo, sabendo que a relação entre galáxias e halos é fundamental para descrever a formação de galáxias e para inferir informação cosmológica a partir das galáxias, nós desenvolvemos uma série de metodologias para obter essa conexão. Usamos métodos de AM diretamente nas propriedades de halos e galáxias, fizemos o mesmo para um conjunto de dados aumentado e usamos ambas as predições para obter modelos empilhados. Convertemos o problema de regressão em classificação, sendo capazes de recuperar não apenas as propriedades das galáxias, mas também sua estocasticidade. Todos esses diferentes projetos focam em aperfeiçoar a obtenção de informação cosmológica de simulações, melhorando sua acurácia e precisão, tanto para os parâmetros cosmológicos, quanto para a conexão galáxia-halo. Estes são os passos iniciais da futura aplicação de metodologias inovadoras em dados reais para a próxima e atual geração de observações.

Palavras-chave: Inferência de parâmetros cosmológicos. Algoritmos de aprendizado de máquina. Simulações cosmológicas. Conexão galáxia-halo.

LIST OF FIGURES

Figure 1 – Pie chart of the relative abundance of matter in the Universe. 34
Figure 2 – Matter evolution of the Universe according to the expansion factor. 41
Figure 3 – Overview of the cosmic history of the Universe in an infographic. 43
Figure 4 – Linear and nonlinear power spectrum comparison. 55
Figure 5 – Dark Matter power spectrum from N-body simulation. 62
Figure 6 – Halo mass function and bias from N-body simulation. 64
Figure 7 – Galaxy stellar mass function comparison. 67
Figure 8 – Examples of underfitting (left panel), balanced model (middle panel), and overfitting (right panel). 71
Figure 9 – Scheme of a decision tree. 74
Figure 10 – Scheme of *mutation* and *crossover* operations into a symbolic regression algorithm. 77
Figure 11 – Representation of a MLP. 78
Figure 12 – Representation of different activation functions. 79
Figure 13 – Visualization of the local patterns learned by convolutional layers. 82
Figure 14 – Scheme of a general spatial transformation of a convolutional layer. 84
Figure 15 – Example of padding operation. 85
Figure 16 – Dropout example. 86
Figure 17 – Example of image denoising. 87
Figure 18 – General scheme of a denoiser auto-encoder. 88
Figure 19 – Example of a graph. 90
Figure 20 – Scheme of an updated graph, after a GNN block, with their node and edge attributes updated. 91
Figure 21 – Halo mass functions comparison. 107
Figure 22 – Slice of the mask of random ellipsoids. 108
Figure 23 – Power spectra comparison. 109
Figure 24 – Comparison of the (normalized) cosmological covariance matrices 114
Figure 25 – Comparison of slices of the normalized covariances. 115
Figure 26 – MSE comparison for the cosmological covariance matrices. 116
Figure 27 – Comparison of the ranked eigenvalues and their relative difference. 117
Figure 28 – Comparison of the diagonal values of the covariance matrices. 118
Figure 29 – Wishart analysis. 118
Figure 30 – Cosmological parameter estimation comparison 121
Figure 31 – A schematic summary of the methodology followed by stacking raw and augmented models. 130
Figure 32 – Augmented distributions by SMOGN 131

Figure 33 – Frequency performance: Raw and SMOGN comparison 132
Figure 34 – Pearson correlation coefficient comparison. 133
Figure 35 – 1D K-S test comparison. 135
Figure 36 – Predicted versus True distributions. 135
Figure 37 – Halo-galaxy property distributions. 136
Figure 38 – Permutation feature importance comparison for the different ML models. . 138
Figure 39 – Permutation feature importance comparison for the stacked raw and SMOGN models. 139
Figure 40 – Power spectra comparison. 140
Figure 41 – Predicted versus true distribution using halo and galaxy information. . . . 144
Figure 42 – Distributions of galaxy properties. 146
Figure 43 – Distribution of halo-galaxy properties 147
Figure 44 – PCC comparison for galaxy properties predicted by Raw, SMOGN, and NNClass. 149
Figure 45 – Examples of graphs constructed from galaxy catalogs from different CAMELS simulations. 158
Figure 46 – Comparison of the number of galaxies per LH catalog in CAMELS simulations. 159
Figure 47 – Ω_m predictions for a model trained on Astrid, using galaxy positions and velocities in the z direction 164
Figure 48 – Ω_m predictions for a model trained on SIMBA and IllustrisTNG, on LH set, using galaxy positions and velocities in the z direction. 166
Figure 49 – Ω_m predictions on SWIFT-EAGLE for a model trained on SIMBA and IllustrisTNG, on LH set, using galaxy positions and velocities in the z direction. 166
Figure 50 – Ω_m predictions for a model trained on SIMBA and IllustrisTNG, on CV set, using galaxy positions and velocities in the z direction. 167
Figure 51 – Ω_m predictions for a model trained on Astrid (with PBC), using galaxy positions and velocities in the z direction and tested considering PBC. . . . 168
Figure 52 – Ω_m predictions for a model trained on Astrid (without PBC), using galaxy positions and velocities in the z direction and tested removing PBC. 169
Figure 53 – Ω_m predictions for a model trained on Astrid, using galaxy positions, velocities in the z direction, and stellar mass. 170
Figure 54 – Ω_m predictions for a model trained on Astrid, using only galaxy positions. 172
Figure 55 – Ω_m predictions for a model trained on Astrid, using only galaxy velocities. 173
Figure 56 – Ratio of the matter power spectrum (left) and global star formation rate density (SFRD) of the 1P simulations (where only one parameter is varied and other parameters are fixed). 175
Figure 57 – Predictions of σ_8 using galaxy and halo properties. 175
Figure 58 – Comparison of Gadget Ω_m predictions for the GNN and SR on halos. 186
Figure 59 – Ω_m predictions using GNNs and SR equations. 187

Figure 60 – GNN (trained on halos) predictions on galaxy catalogs. 188
Figure 61 – Tuned SR equations (trained on GNN blocks from halo catalogs) predictions on galaxy catalogs. 190
Figure 62 – Examples of 2D graphs built from galaxy catalogs from different CAMELS simulations: Astrid, SIMBA, IllustrisTNG, Magneticum, SB28, and SWIFT-EAGLE. 197
Figure 63 – Averaged predictions over Astrid, SIMBA, IllustrisTNG, SB28, Magneticum, and SWIFT-EAGLE for the different observational effects compared to the case of disregarding systematics. 199

LIST OF TABLES

Table 1 – Bias for the halo catalogs. 110
Table 2 – Bias in the estimated parameters 122
Table 3 – MSE and PCC scores. 134
Table 4 – 2D K-S test comparison. 137
Table 5 – Bin edges and central values for the subsets considered for the computation of the power spectrum. 140
Table 6 – K-S test values for univariate (1D) and joint (2D) distributions computed with the NNs and the baseline models. 148
Table 7 – Characteristics of the hydrodynamical simulations. 153
Table 8 – Analytic formulae obtained using SR for each component of the learned GNN model. 185
Table 9 – List of the optimized values of δ. 189
Table 10 – Values of the linking radius found by Optuna for the selected models. . . . 196

LIST OF ABBREVIATIONS AND ACRONYMS

AGN	Active Galactic Nuclei
AI	Artificial Intelligence
BBKS	Bardeen, Bond, Kaiser and Szalay
BAO	Baryon Acoustic Oscillations
BH	Black Holes
CIC	Cloud In Cell
CNNs	Convolutional Neural Networks
CMB	Cosmic Microwave Background
DE	Dark Energy
DM	Dark Matter
DTs	Decision Trees
DL	Deep Learning
ERT	Extremely Randomized Trees
FFT	Fast Fourier Transform
FKP	Feldman, Kaiser, and Peacock
FI	Feature Importance
FLRW	Friedman-Lemaître-Robertson-Walker
FoF	Friend-of-Friends
GR	General Relativity
GBDTs	Gradient Boosting Decision Trees
GCN	Graph Convolutional Network
GNNs	Graph Neural Networks
HOD	Halo Occupation Distribution
IG	Information Gain

ICs	Initial Conditions
kNN	k Nearest Neighbors
KDE	Kernel Density Estimators
K-S	Kolmogorov-Smirnov
ΛCDM	Lambda Cold Dark Matter
LPT	Lagrangian Perturbation Theory
LGBM	Light Gradient Boosting Machine
ML	Machine Learning
MCMC	Markov Chain Monte Carlo
MNIST	Modified National Institute of Standards and Technology
MAE	Mean Absolute Error
MSE	Mean Squared Error
MNNs	Moment Neural Networks
MLPs	Multi Layer Perceptrons
NN	Neural Networks
PM	Particle Mesh
RF	Random Forests
ReLU	Rectified Linear Unit
RGB	Red, Green, and Blue
REDNet	Residual Encoder-Decoder Network
SAMs	Semi-Analytical Models
sSFR	specific Star Formation Rate
SO	Spherical Overdensity
SR	Symbolic Regression
SMOTe	Synthetic Minority Over-sampling Technique
SMOGN	Synthetic Minority Over-sampling Technique for Regression with Gaussian Noise

SFR	Star Formation Rate
SHAM	Subhalo Abundance Matching
SN	Supernova
TPE	Tree Parzen Estimator
UV	Ultra-Violet

CONTENTS

1 INTRODUCTION . . . 33
1.1 Goals and organization of the thesis . . . 36

2 COSMOLOGY BACKGROUND . . . 39
2.1 From the beginning: Classical Cosmology . . . 39
2.2 The history of the Universe in a nutshell . . . 42
2.3 Cosmology Toolbox . . . 45
2.3.1 Transforming structure formation into structure information . . . 45
2.3.2 Linking observations to theory . . . 46
2.3.3 Computing the power spectrum . . . 47
2.3.4 The number density of objects and the halo model . . . 47
2.3.5 Bayesian statistics, parameter inference, and cosmological covariance matrices 48
2.3.5.1 Covariance matrices . . . 49
2.4 Cosmology in the linear regime . . . 50
2.5 Nonlinear Cosmology . . . 55
2.5.1 N-body simulations . . . 56
2.5.1.1 Implementation stage: solving the equations . . . 58
2.5.1.2 The density field . . . 58
2.5.1.3 The density contrast field and its Fourier transform . . . 59
2.5.1.4 The gravitational potential . . . 59
2.5.1.5 The acceleration . . . 60
2.5.1.6 Updating particles positions and momenta . . . 60
2.5.1.7 N-body power spectrum . . . 61
2.5.1.8 N-body halo finder, mass function, and bias . . . 62
2.5.1.9 Approximated methods for DM simulations . . . 64
2.5.2 Hydrodynamical simulations . . . 64
2.5.2.1 Approximated methods to galaxies and halo-galaxy connection . . . 67

3 MACHINE LEARNING BACKGROUND . . . 69
3.1 From the beginning: machine learning notions . . . 69
3.2 Traditional Machine Learning Methods . . . 73
3.2.1 k-Nearest Neighbors . . . 73
3.2.2 Tree methods . . . 73
3.2.2.1 Extreme Randomized Trees . . . 75
3.2.2.2 Light Gradient Boosting Machines . . . 76
3.2.3 Symbolic Regression . . . 76

3.3 Deep Learning Methods . . . 78
3.3.1 Multi Layer Perceptrons . . . 78
3.3.2 Convolutional Neural Networks . . . 82
3.3.2.1 Convolutional Neural Network Blocks . . . 82
3.3.2.1.1 Convolutional Layers . . . 82
3.3.2.1.2 Pooling Layers . . . 85
3.3.2.1.3 Dropout Layers . . . 86
3.3.3 Image denoising techniques . . . 87
3.3.4 Graph Neural Networks . . . 89
3.3.4.1 Graphs . . . 90
3.3.4.2 Graph Neural Network Blocks . . . 91
3.3.4.2.1 Meta Layer . . . 92
3.3.4.2.2 Graph Convolutional Layers . . . 93
3.3.4.2.3 SAGE Convolutional Layer . . . 93
3.3.4.3 GNN variations . . . 94
3.3.4.3.1 Deep sets . . . 95
3.3.4.3.2 No initial node attributes . . . 95
3.4 Probabilistic Methods . . . 96
3.4.1 Moment Neural Networks . . . 96
3.4.2 Regression to Classification: NNClass . . . 97
3.5 Other Machine Learning Tools . . . 97
3.5.1 The problem of Imbalanced Data Sets . . . 98
3.5.2 Combining different methods . . . 99
3.5.3 Feature Importance . . . 99
3.5.4 Hyperparameter search . . . 100

4 IMPROVING COSMOLOGICAL COVARIANCE MATRICES . . . 103
4.1 Motivation . . . 104
4.2 Halo catalogs . . . 106
4.2.1 ExSHalos . . . 106
4.2.2 Quijote . . . 106
4.2.3 The match between the catalogs . . . 107
4.3 The data set of covariance matrices . . . 108
4.3.1 The power spectrum . . . 108
4.3.2 The bias . . . 110
4.3.3 The covariance matrices . . . 110
4.4 The ML suite . . . 111
4.5 Results . . . 113
4.5.1 Visualizing the matrices . . . 113
4.5.2 The MSE between different matrices . . . 115

4.5.3 The eigenvalues of the matrices . . . 116
4.5.4 The diagonal values of the matrices . . . 117
4.5.5 An analytical comparison for the machine learning black box . . . 118
4.5.6 Recovering the cosmological parameters . . . 120
4.6 Discussion and Conclusions . . . 122

5 MIMICKING THE HALO-GALAXY CONNECTION . . . 125
5.1 The IllustrisTNG data . . . 126
5.1.1 Halo Properties . . . 127
5.1.2 Galaxy Properties . . . 128
5.1.3 Data pre-selection . . . 128
5.2 Performance Metrics . . . 129
5.3 Stacking raw and augmented models . . . 129
5.3.1 The SMOGN galaxy distributions . . . 130
5.3.2 Results . . . 131
5.3.2.1 Frequency performance . . . 131
5.3.2.2 Metric performances . . . 133
5.3.2.2.1 1D K-S test . . . 134
5.3.2.3 Predicted versus True distributions . . . 134
5.3.2.4 Halo-galaxy property distributions and the 2D K-S test . . . 136
5.3.2.5 Feature Importance Analysis . . . 137
5.3.2.6 Power Spectrum . . . 139
5.3.3 Discussion and conclusions . . . 141
5.4 Converting regression to classification . . . 145
5.4.1 Methodology . . . 145
5.4.2 Results . . . 145
5.4.2.1 Distribution of halo-galaxy properties . . . 145
5.4.2.2 K-S test for galaxy predictions . . . 148
5.4.2.3 Single Value Estimation . . . 148
5.4.3 Discussion and conclusions . . . 149

6 FIELD-LEVEL LIKELIHOOD-FREE INFERENCE WITH GRAPH NEURAL NETWORKS . . . 151
6.1 Robust field-level likelihood-free inference with galaxies . . . 152
6.1.1 Data . . . 153
6.1.1.1 Simulations . . . 153
6.1.1.2 Galaxy catalogs . . . 155
6.1.1.3 Galaxy graphs: construction . . . 156
6.1.2 GNN architecture . . . 159
6.1.2.1 Variations of the architecture . . . 160

6.1.3 Likelihood-free inference and the loss function . . . 161
6.1.4 Training procedure and optimization . . . 161
6.1.5 Performance Metrics . . . 162
6.1.6 Results . . . 163
6.1.6.1 Positions & velocities . . . 163
6.1.6.1.1 Astrid results . . . 163
6.1.6.1.2 SIMBA and IllustrisTNG results . . . 166
6.1.6.1.3 Super-sample covariance analysis . . . 167
6.1.6.2 Positions, velocities, and stellar mass . . . 169
6.1.6.3 Where does the information come from? . . . 171
6.1.6.3.1 Why the model trained on Astrid is so good? . . . 174
6.1.7 Why not inferring σ_8? . . . 174
6.1.8 Discussion and conclusions . . . 176
6.2 A universal equation to predict Ω_m from halo and galaxy catalogs . . . 179
6.2.1 Data . . . 179
6.2.1.1 Halo and galaxy catalogs . . . 180
6.2.2 GNN architecture . . . 181
6.2.2.1 Training procedure and optimization . . . 182
6.2.3 SR architecture . . . 183
6.2.4 Results . . . 185
6.2.4.1 Analytic Approximations . . . 185
6.2.4.2 Predictions on halo catalogs . . . 187
6.2.4.3 Predictions on galaxy catalogs . . . 189
6.2.5 Discussion and Conclusions . . . 192
6.3 The impact of systematic effects . . . 193
6.3.1 Data . . . 194
6.3.1.1 Galaxy graphs . . . 196
6.3.2 Methodology . . . 198
6.3.3 Results . . . 198
6.3.4 Discussion and conclusions . . . 200

7 DISCUSSION AND CONCLUSIONS . . . 201

REFERENCES . . . 205

APPENDIX 227

APPENDIX A – SOLVING THE POISSON EQUATION IN FOURIER SPACE . . . 229

1 INTRODUCTION

According to P. J. E. Peebles, as outlined in Principles of Physical Cosmology (1), we can define:

"*Physical cosmology is the attempt to make sense of the large-scale nature of the material world around us, by the methods of natural sciences*".

In this context, Cosmology is the area of science responsible for describing how our Universe works, elaborating on how to the components present in the very early Universe evolved into the structures we can observe today (2–4). This is not an easy task, which is why Cosmology makes use of many other fields, such as Astronomy, General Relativity, Particle Physics, Quantum Field Theory, Physical Statistics, and even Computer Science, developing alongside them. Simultaneously, the abundance of observational data provides us with the means to carefully select from the complete theoretical and experimental framework developed thus far. This access to a vast array of observations enables us to make informed decisions in both current and future cosmological research.

The birth of Cosmology was given by the development of astronomical observations together with General Relativity (GR). GR, formulated by Albert Einstein between 1914 and 1917, introduced the well-known Einstein's equations. These equations were crafted to elucidate the gravitational force and to position the Universe as a central subject of study (5). Subsequently, from 1917 to 1922, the Friedmann-Robertson-Leimaître-Walker metric was developed (6–9), describing the evolution of the Universe as homogeneous and isotropic. Some years later (1927-1929), observations related to the correlation between distances of galaxies and their velocities confirmed that the Universe is indeed expanding, with Lemaître and Hubble being instrumental in this confirmation (10, 11). Ideas regarding the origin of the Universe began to take shape in subsequent years, notably with Lemaître in 1931 (8). However, the comprehensive theory of the Big Bang only fully emerged in 1948 (12). A pivotal component of the Big Bang theory, claiming that the Universe expanded from a hot dense state dominated by thermal black body radiation, is the observable cosmic microwave background (CMB). Penzias and Wilson observed the CMB in 1965, marking a significant milestone in cosmological research (13).

The final product of Cosmology nowadays is the ΛCDM (Lambda cold dark matter) model, which has been supported over the past years by various observations. These observations include anisotropies of the CMB (14), rotational curves of galaxies leading to the identification of Dark Matter (DM) (15, 16), and observations of distant supernovae Ia, which not only reveal the accelerated expansion of the Universe but also hint at the presence of Dark Energy (DE) (17, 18). Notably, DE can be interpreted as the cosmological constant Λ, introduced by Einstein in 1917 (5). According to the ΛCDM model, the Universe is composed of three main

components: Λ, representing DE; a cold (non-relativistic) type of matter that only interacts gravitationally, i.e., DM; and ordinary matter (or baryonic matter). These components constitute approximately 68.5%, 26.5%, and 5% of the Universe, respectively. A minor fraction ($\lesssim 0.002\%$) of the overall content is allocated to radiation, encompassing photons, CMB, gravitons, and neutrinos. Figure 1 illustrates a pie chart depicting the matter distribution in the Universe. It is worth noting that DM and baryonic matter are represented with the same color, as they are both part of the matter component.

Despite its notable success, the ΛCDM model fails to explain the nature of both DM and DE, among other unresolved questions. These questions extend to issues such as tensions in certain cosmological parameters like the Hubble parameter (H_0) or the relationship between the matter content (Ω_m) and the power spectrum amplitude at $8h^{-1}$ Mpc (σ_8), denoted as $S_8 = \sigma_8 \left(\Omega_m/0.3\right)^{1/2}$ (19–21). Given the presence of these persistent puzzles and tensions, our current objective is to rigorously constrain these cosmological parameters, aiming for the highest possible accuracy in our pursuit of understanding the fundamental aspects of the Universe.

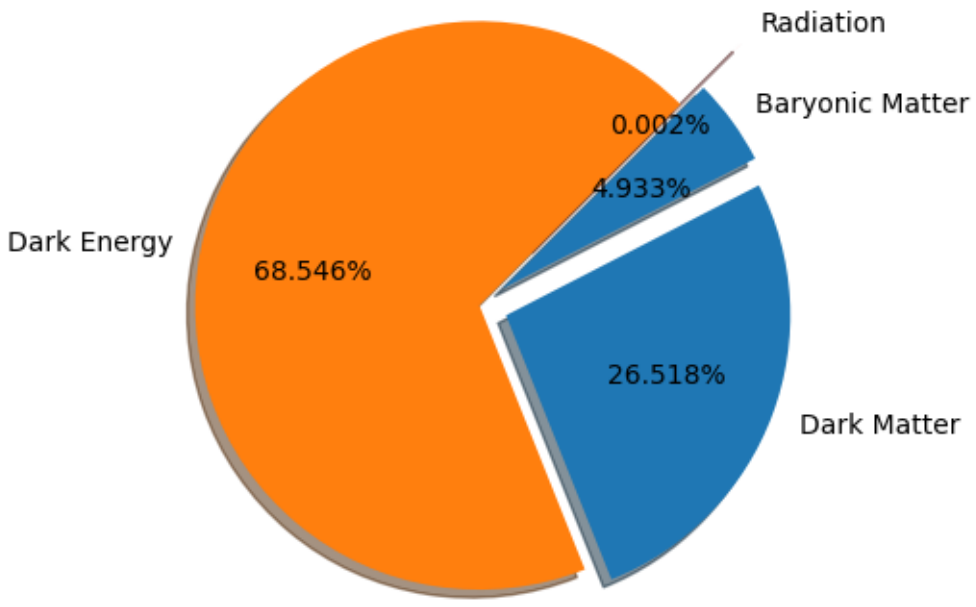

Figure 1 – **Pie chart of the relative abundance of matter in the Universe.** We take into account dark energy, radiation, baryonic matter, and dark matter.

Since the distribution of matter and galaxies in the Universe depends on the cosmological parameters, the clustering of these objects can be used to infer the values of those parameters. To gather as much diverse data as possible, large international efforts are currently underway to survey the cosmos at different wavelengths. These include projects such as DESI (22), Euclid (23–26), PFS (27), J-PAS (28), SKA (29), Roman (30), JWST (31), and others. The data

collected from these missions will encompass larger volumes at different redshifts, observing a variety of galaxy types across many wavelengths. Extracting the maximum amount of relevant information from these data sets is crucial in improving our understanding of fundamental physics.

To achieve this objective, theoretical predictions and methods for extracting information are necessary. On the one hand, we have traditional methods for extracting information from cosmological observations. In the case of Bayesian inference schemes for cosmological parameters, nearly all analyses rely on summary statistics, such as the power spectrum. However, this approach is sub-optimal as we do not know which summary statistics contain all (or the majority) of the cosmological information across all scales (32–35). Furthermore, conventional methods typically require costly simulations to either estimate covariance matrices or to forward-model the observations (36–40).

On the other hand, machine learning (ML) techniques have demonstrated superiority over traditional methods in various contexts and fields, including Cosmology and Astrophysics. The power of these new methods lies precisely in their capacity to handle large and complex data sets, providing nonlinear relations in high-dimensional feature spaces that enable solving regression and classification tasks (41). By using different summary statistics as input data, authors from Reference (42) were able to derive cosmological parameters without the need for additional input from theoretical models. This represents a potent extension of traditional Monte Carlo-based methods. Particularly, likelihood-free inference methods (also known as simulation-based inference or implicit likelihood inference) utilize forward models of observables instead of analytic models. Subsequently, these methods infer a posterior distribution over the parameters (43). Several studies have demonstrated their competitiveness with traditional statistical inference methods (44–52). At a level closer to the observations and simulations, numerous studies exploring the halo-galaxy connection can make predictions comparable to the output of numerical/analytical methods (53–69). An added advantage of ML models is that, once trained, they typically make predictions much faster than traditional methods (66). However, a notable disadvantage arises when these models struggle to extrapolate predictions across different data sets than those with which they were initially trained (48, 50, 70).

The abundance of data combined with a rapidly expanding menu of different methods represents a challenge for our understanding of the structure formation and evolution of the Universe. The overarching goal is to extract the maximum information from ongoing and forthcoming surveys, as well as from the state-of-art of simulations and approximated methods, in order to select the most fitting models. Ultimately, we aim to delve into the nature of the cosmos and all its constituent elements. This collective pursuit represents a crucial step toward unraveling the mysteries that the Universe presents, fostering a deeper understanding through the synthesis of observational, theoretical, and computational approaches.

1.1 Goals and organization of the thesis

The primary focus of this thesis revolves around the application of ML techniques to extract cosmological information. Each sub-project within this thesis is dedicated to developing a robust ML method and provide predictions with the highest accuracy and precision. The objective is not to replace traditional methods but to offer as a viable alternative to do Cosmology nowadays. Importantly, each sub-project has culminated in scientific publications. The sub-projects are aligned with three main approaches:

- **Improving Cosmological Covariance Matrices.** Cosmological covariance matrices play a pivotal role for parameter inference using Markov Chain Monte Carlo (MCMC) explorations of the likelihood. This methodology involves comparing the theoretical model with a summary statistic taken over many realizations of a simulation or an approximated method. The precision and accuracy of this pipeline depend on the number of realizations of the statistic used to construct a cosmological covariance matrix. Realistically, running thousands of hydrodynamical, N-body, or even approximated simulations to obtain these matrices is often impractical. In the project presented in the Chapter 4, we explored the capability of an image denoising technique to take a cosmological covariance matrix constructed with only hundreds of spectra and generate a matrix resembling one created with thousands of spectra. Moreover, we trained our algorithm using only approximated simulations (which are fast to run) and were able to extrapolate their predictions for matrices built with realistic N-body simulations. Utilizing the Wishart distribution, we demonstrated that the denoiser's end product can be compared to an effective sample augmentation in the input matrices (40). This work serves as a proof of concept that we can borrow computer vision techniques and apply them to improve even traditional approaches in Cosmology.

- **Mimicking the halo-galaxy connection.** The relationship between galaxies and halos is central to the description of galaxy formation and is a fundamental step towards extracting precise cosmological information from galaxy maps. However, this connection involves several complex and interconnected processes. By utilizing halo properties such as halo mass, concentration, spin, and halo overdensity, we are obtaining central galaxy properties such as stellar mass, specific star formation rate (sSFR), color, and size, as outlined in Chapter 5. During this process we have followed two different approaches:

    1. *Employing traditional ML methods, namely extremely randomized trees (ERT), k-nearest neighbors (kNN), light gradient boosting machine (LGBM), and neural networks (NN)*[1]. Rather than selecting the best model, we adopted a stacking approach, combining their predictions to derive a single prediction for each galaxy property. The

[1] Note that, here, we are referring as NN a Multi Layer Perceptron (MLP).

ensemble technique aimed to enhance the overall predictive performance beyond what individual models could achieve. However, we observed that the inherent stochasticity in galaxy properties was not faithfully reproduced, as ML models often focused on the peaks of the distributions. Recognizing this challenge, we framed the issue as an imbalanced data set problem and applied the Synthetic Minority Over-sampling Technique for Regression with Gaussian Noise (SMOGN), in the galaxy properties, to address it. While this technique proved effective in alleviating the problem, achieving a perfect reproduction of the complete scatter in galaxy properties remained elusive (65). Nonetheless, this work represents a significant step towards refining the modeling of the intricate relationship between halos and galaxies in the context of ML.

2. *Transforming a regression to a classification problem: NNClass.* To address the scatter problem for galaxy predictions, we transformed them from a single value to a probability density distribution for each galaxy property (one at a time, or two simultaneously) given a set of halo attributes. To achieve this, we employed a NN[2] and altered the activation function of the last layer from linear activation to a SoftMax, and the loss function from mean squared error to Cross-Entropy. The outcome of this approach demonstrated its effectiveness in recovering the complete scatter over the 2D galaxy-galaxy and galaxy-halo properties for all galaxies. By adopting a probabilistic framework for the halo-galaxy connection, as introduced in Reference (69), we opened up new avenues for understanding and modeling the complex relationships between halos and galaxies.

- **Field-level Likelihood-free Inference with Graph Neural Networks.** In this work we have tailored the Graph Neural Networks (GNNs), together with the Moment Neural Networks (MNNs), to get the first robust suite over five different codes and subgrid models – IllustrisTNG, SIMBA, Astrid, Magneticum, SWIFT-EAGLE – changes in astrophysics, and subhalo-galaxy finder. The models are rotational, translational, and permutation invariant and do not impose any cut on scale, using only galaxy positions and v_z velocities to infer Ω_m (71). We also have shown:

  1. We are able to translate the GNN operations into analytical equations, using symbolic regression (SR). This gave us some insights over the physical relation between the use of objects positions and velocities, speculating that the suite was learning a physical relation between them and the matter content of the simulations, instead of a trivial correlation (72).
  2. Because the best subgrid physical model used as data set to train the GNNs was Astrid, we have started to analyze their differences with the other hydrodynamical models. The main conclusions were related to the broader variations on their galaxy

[2] Note that, once again, we are referring as NN a Multi Layer Perceptron (MLP).

properties and their number per different variations through cosmological and astrophysical parameters (71, 73).

3. The next phase of our exploration involved speculating on the applicability of our suite to real galaxy catalogs. The initial step in this direction involved incorporating observational effects into galaxy catalogs derived from hydrodynamical simulations. We considered various factors, including: (i) mask effects, (ii) uncertainties in peculiar velocities and radial distances, and (iii) different galaxy selections. Despite the increased level of complexity introduced by these observational effects, our suite demonstrated robust performance. In at least 90% of the available data, the model exhibited strong predictive capabilities. This encouraging result emphasized the potential of our methodology to real observational data. Detailed findings and implications were documented in Reference (74), showcasing the promise of our approach in handling and extracting valuable information from real-world galaxy catalogs.

This thesis is then organized according to: first, we make a general presentation of the whole set of methods we have used, from Cosmology (in Chapter 2) to Machine Learning techniques (in Chapter 3); second, we present the improvements to the cosmological covariance matrices (in Chapter 4); third, we discuss the halo-galaxy connection in the context of ML applications (in Chapter 5); fourth, we present the area of field-level likelihood-free inference with graph neural networks (in Chapter 6); then, we finalize with a discussion, conclusions, and next steps for all the work developed along these years.

2 COSMOLOGY BACKGROUND

This thesis is built upon the principles of Cosmology. In this chapter, you will find a brief overview of it, which will serve as a basis for the next chapters. For a more comprehensive understanding of the topics discussed here, we recommend consulting References (1–3, 75, 76).

Firstly, in Section 2.1, we offer an overview of ΛCDM model, beginning with Classical Cosmology. This encompasses the Friedman-Lemaître-Robertson-Walker (FLRW) metric, Friedman equations, the different eras of the Universe, and culminates with a discussion on cosmological parameters, the estimation of which forms a primary objective of many projects developed within this thesis. Secondly, we provide a brief introduction to the history of the observable Universe, in Section 2.2. Thirdly, in Section 2.3, we present an overview of traditional cosmological methods, including discussions on the power spectrum, correlation function, bias, elements of the halo model, Bayesian statistics, and sample covariance matrices. Fourthly, we introduce linear theory, in Section 2.4, providing motivation for the presentation of the measured linear power spectrum. This is followed by Section 2.5, where we discuss the motivation for studying the nonlinear power spectrum while describing nonlinear theory through N-body and hydrodynamic simulations.

2.1 From the beginning: Classical Cosmology

Classical Cosmology begins with GR (77–79). At the heart of GR are Einstein's equations:

$$G_{\mu\nu} \equiv R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} = 8\pi G T_{\mu\nu}, \tag{2.1}$$

where $g_{\mu\nu}$ is the metric tensor, $R_{\mu\nu}$ is the Ricci tensor, R is their trace, and $T_{\mu\nu}$ is the energy-momentum tensor. The left side of these equation represent the geometric component of the Universe, while the right side represents the source of energy and momentum that acts as the source of the curvature of the space-time.

All the geometric components came from the metric tensor by definition:

$$R_{\mu\nu} \equiv \partial_\rho \Gamma^\rho_{\mu\nu} - \partial_\nu \Gamma^\rho_{\mu\rho} + \Gamma^\rho_{\rho\lambda}\Gamma^\lambda_{\mu\nu} - \Gamma^\rho_{\nu\lambda}\Gamma^\lambda_{\rho\mu}, \tag{2.2}$$

$$R \equiv g^{\mu\nu} R_{\mu\nu} = R^\mu{}_\mu, \tag{2.3}$$

$$\Gamma^\alpha_{\beta\gamma} \equiv \frac{g^{\alpha\delta}}{2} \left(\partial_\beta g_{\delta\gamma} + \partial_\gamma g_{\beta\delta} - \partial_\delta g_{\beta\gamma} \right). \tag{2.4}$$

where $g^{\mu\nu} g_{\nu\sigma} = g_{\lambda\sigma} g^{\lambda\mu} = \delta^\mu_\sigma$ or $g^{\mu\nu} g_{\mu\nu} = \mathbb{I}$, and $\Gamma^\alpha_{\beta\gamma}$ is the Christoffel symbol. The energy-momentum tensor $T_{\mu\nu}$, in general, takes a complicated form. However, considering a *perfect fluid*, which can be completely characterized by its pressure p and energy density ρ, we can write

$$T_{\mu\nu} = (\rho + p)\, u_\mu u_\nu + p g_{\mu\nu}, \tag{2.5}$$

where u^μ is the fluid four-velocity.

Solving Einstein's equations involves determining the metric tensor. However, these equations are nonlinear, which represents a challenge for finding their solutions. Nonetheless, exploiting the symmetries of space-time can simplify the process and make the path to their solutions more straightforward.

This can be done in the cosmological framework considering the Universe as homogeneous, isotropic, and temporally evolving with time. This is translated by the FLRW metric (6–9, 77–79), which can be written as

$$ds^2 = -dt^2 + a^2(t)\left[\frac{dr^2}{(1-kr^2)} + r^2\left(d\theta^2 + \sin^2\theta\, d\phi^2\right)\right], \tag{2.6}$$

where $a(t)$ is the expansion factor, r is the radial coordinate, θ and ϕ are the angular coordinates, and k is the curvature parameter[1], with dimensions of L^{-2}.

Considering a *perfect fluid* (see Equation 2.5) and plugging this metric into Einstein's equations, we obtain the *Friedmann equations*, which define the temporal evolution of matter and energy as

$$\left(\frac{\dot{a}}{a}\right)^2 = H^2(t) = \frac{8\pi G}{3}\rho - \frac{k}{a^2}, \tag{2.7}$$

$$\frac{\ddot{a}}{a} = -\frac{4\pi G}{3}\left(\rho + 3p\right), \tag{2.8}$$

with $\dot{a} = \frac{da}{dt} = H(t)$, $H(t)$ the Hubble parameter (which determines how fast the Universe expands), ρ represents the amount of matter considered, and p denotes the pressure. To measure distances and time (since we are defining $c \equiv 1$), we often use the concept of *redshift* z. The redshift is related to the expansion factor according to $a = 1/(1+z)$.

From the Friedmann equations, we can derive the *continuity equation*, which states the conservation of the total energy

$$\frac{d\rho}{dt} + 3H\left(1+\omega\right)\rho = 0, \tag{2.9}$$

where

$$\omega = \frac{p}{\rho} \tag{2.10}$$

represent what we call as *equation of state*. The total amount of matter correspond to the sum over the different components of the energy content in the Universe: $\rho = \sum_i \rho_i$, where i refers to *radiation* (γ), *matter* (m), and *dark energy* (Λ). Radiation include photons, cosmic background radiation, gravitons, and neutrinos, i.e., particles satisfying $p_\gamma = \rho_\gamma/3$. *Matter* refers to the baryonic matter and *dark matter*, the later being a type of matter that does not interact with

[1] The curvature parameter k classifies the Universe as follows: closed with positive curvature ($k = 1$), plane without curvature ($k = 0$), and open with negative curvature ($k = -1$).

electromagnetic radiation and has negligible pressure compared to its density: $p_m = 0$. Finally, we have *dark energy*, an unknown form of energy that uniformly fills the Universe and mimics *vacuum energy*. It is often regarded as the *cosmological constant* Λ, with the equation of state given by $p_\Lambda = -\rho_\Lambda$.

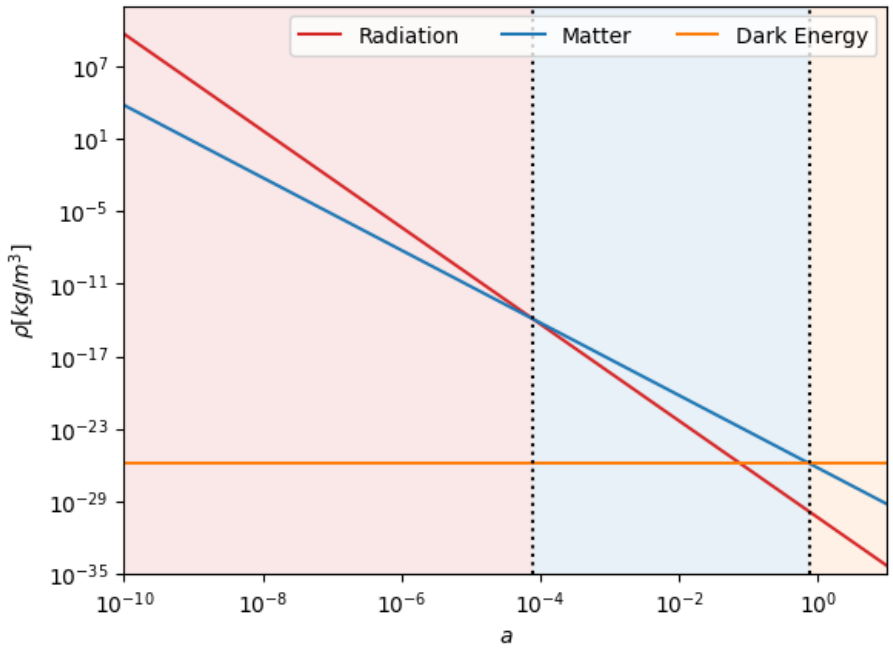

Figure 2 – **Matter evolution of the Universe according to the expansion factor a.** We represent by the colored shadow regions the phase of domination for each content, where the vertical dotted lines correspond to the intersection point of domination between the different eras.

Returning to Equation 2.7, we observe that the expansion of the Universe is determined by the contributions from the different components of matter and energy. Firstly, we define the *critical density*, denoted by $\rho_{crit}(t)$, by considering the density of a flat Universe ($k = 0$), which is experimentally confirmed through observational measurements such as those found by Planck (14). Thus, we have

$$\rho_{crit}(t) = \frac{3H^2(t)}{8\pi G} \quad \text{and} \quad \rho_{crit,0} = \frac{3H_0^2}{8\pi G}, \tag{2.11}$$

where $\rho_{crit,0}$ is its value today, defined according to the Hubble parameter today H_0. Secondly, we define the cosmological parameters related to the amount of each matter component in the Universe as

$$\Omega_i(t) = \frac{\rho_i(t)}{\rho_{crit}(t)} \quad \text{and} \quad \Omega_i = \frac{\rho_{i,0}}{\rho_{crit,0}}, \tag{2.12}$$

where, again, i correspond to matter (m), radiation (γ), and dark energy (Λ). Thirdly, noting that we can combine Equations 2.9 and 2.10, we can express how each component evolves with

the scale factor as

$$\rho_\gamma = \rho_{\gamma,0} a^{-4}, \tag{2.13}$$

$$\rho_m = \rho_{m,0} a^{-3}, \tag{2.14}$$

$$\rho_\Lambda = \rho_{\Lambda,0}. \tag{2.15}$$

Therefore, finally returning to Equation 2.7, we obtain

$$H^2(t) = H_0^2 \left[\Omega_\gamma a^{-4}(t) + \Omega_m a^{-3}(t) + \Omega_\Lambda\right]. \tag{2.16}$$

In addition to describing the evolution of the Hubble parameter (see Equation 2.16), the components of the Universe lead to another interesting property: their time-dependence, which can be expressed in terms of the scale factor a, the redshift z, or time t. This dependence defines what we call the matter evolution of the Universe or its cosmic timeline, which can be divided into different eras according to the dominant matter contribution. By utilizing Equations 2.13, 2.14, 2.15, along with the current values for $\Omega_m = 0.315$, $\Omega_\gamma = 2.473 \cdot 10^{-5}$, and $\Omega_\Lambda = 0.685$, as in Reference (80), we can visualize the matter evolution of the Universe in Figure 2. In the primordial Universe, the scale factor dependence $\propto a^{-4}$ is the largest one, corresponding to the *radiation era*. This persists up to $\sim 10^{-4}$ when matter starts to dominate with $\propto a^{-3}$ in the *matter era*. Finally, as a approaches 1, we enter the current era, the *dark energy era*, where the amount of matter evolves constantly with the scale factor.

Today, the ΛCDM model stands as the most successful model in Cosmology, described by only 6 independent parameters: $\Omega_b h^2$, $\Omega_c h^2$, Θ_s, τ_T, n_s, and A_s (14, 81). Here, Ω_b and Ω_c represent the amount of baryons and cold DM, respectively, with $\Omega_m = \Omega_b + \Omega_c$. h is the dimensionless Hubble parameter: $H_0 = 100h$ km s^{-1} Mpc^{-1}, which has become the focus of attention due to the so-called Hubble tension (82) that confronts the supernovae measurements (19) and the CMB inference for that parameter (14). Θ_s is the angle given by the ratio between the sound horizon r_s and the angular diameter distance at decoupling. τ_T represents for the optical depth. Meanwhile, n_s (scalar spectral index) and A_s (amplitude of the primordial spectrum) are related to the features of the primordial spectrum of density fluctuations (presumably from inflation). In principle, all the other mentioned parameters can be derived from the ones presented here.

2.2 The history of the Universe in a nutshell

We have already been through a brief history of the Universe whilst using Classical Cosmology to differentiate between the different eras. Now it is time to delve deeper into this journey, specifying some details about the relevant epochs and the transitions between them. A pictorial version of cosmic history can be found in Figure 3, and its main cornerstones are as follows:

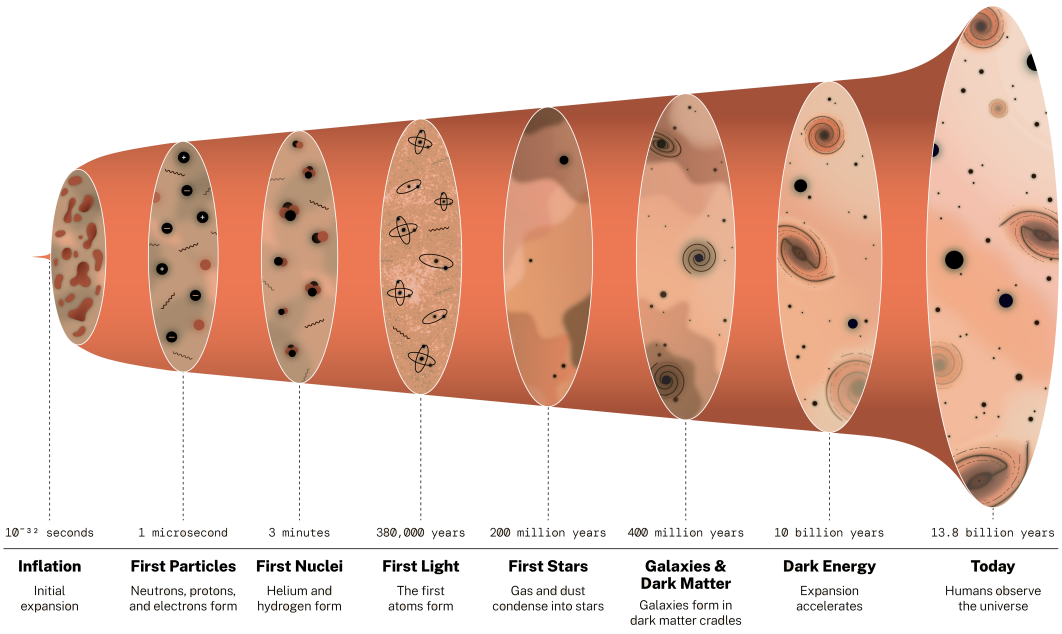

Figure 3 – **Overview of the cosmic history of the Universe in an infographic.** *Source:* NASA (83).

- **The early Universe: Big Bang, Inflation, and Radiation Era.** As we have already seen, it is believed that the Universe originated from an event called the *Big Bang* (12). The subsequent epoch, many believe, is called *inflation*, representing the phase during which the Universe began a period of accelerated expansion: $a \sim e^{H_{inf}t}$, where H_{inf} is the Hubble parameter during inflation. Throughout the inflationary period (until 10^{-32} seconds), the energy content of the Universe was dominated by a scalar field. The small curvature (and density) fluctuations produced during this period created the seeds for the structures we can observe today (84, 85). At the end of inflation this field decayed into other particles and radiation, initiating the *radiation era*, when the Universe entered a phase where it could be very well described by a hot, dense, and homogeneous plasma in thermal equilibrium.

- **Primordial Nucleosynthesis and Matter Era.** Following this equilibrium, the expansion of the Universe, together with the resulting cooling down, led to the progressive decoupling of some components from the cosmic plasma. At about 3 minutes after the Big Bang, protons and neutrons were cold enough that they could combine and produce the earliest light elements–hydrogen and helium (86, 87). Following this phase of primordial nucleosynthesis, around 10^4 years after the Big Bang, due to the mentioned scaling dependencies (see Equations 2.13, 2.14, 2.15), we reach the matter-radiation equality, where matter becomes equally important as radiation. After that point, radiation is still important but the Universe's expansion is dominated by non-relativistic species, which marks the beginning of the *matter era*.

- **Recombination and CMB decoupling.** Two fundamental processes occurred at $\sim 3.8 \cdot 10^5$ years into the history of the Universe. The first is called *recombination*, which resulted from the progressive cooling of the cosmic plasma, to the point where photons do not have the energy required to keep electrons ionized from the positively charged nuclei anymore, resulting in the formation of the first *neutral atoms* (88). Small perturbations in the fluid of baryons and photons then began to propagate as *acoustic waves*, characterized by their sound speed c_s and sound horizon r_s. This last parameter corresponds to the maximum distance that acoustic waves could have traveled since the Big Bang up to that time, given their sound speed and the expansion history up to that point. The sound horizon is the origin of the *baryon acoustic oscillations* (BAO), a characteristic length scale imprinted on all density perturbations which is a key observable that modern galaxy surveys seek to detect (89). The second process is the *CMB decoupling*: after recombination, the decreasing number of free electrons reduced the rate of interactions between photons and the baryons of the cosmic plasma, which kept the baryonic matter coupled to radiation. As a result, photons decoupled from baryons, evolving separately thereafter. The released photons now form what we observe as the CMB, representing the oldest light observable in the Universe (*the first light*) (90).

- **Dark ages, the First Stars, and the First Galaxies.** From the appearance of the CMB until $\sim 2 \cdot 10^8$ years, the Universe was basically neutral (no free electrons apart from a small fraction left out by recombination). Due to the absence of light sources at that time, this epoch is called the *dark ages* – a period that is particularly challenging to observe, apart from possibly 21cm observations of neutral hydrogen at extremely high redshifts (91). However, during this time the growth of structures was already becoming significant: inhomogeneities were being magnified, leading to the collapse of matter into the first DM halos. In other words, at this time the Universe witnessed the emergence of the first large-scale structures. From $2 \cdot 10^8$ to $4 \cdot 10^8$ years, gas and dust began to condense into stars within these massive halos, initiating the formation of the most ancient galaxies (and kick-starting the complex relationships involved in the halo-galaxy connection). These first light sources emitted ultra-violet (UV) and X-ray radiation, potentially leaving imprints in the 21cm observations of that era as well.

- **Reionization and Dark Energy Era.** From approximately $\sim 5 \cdot 10^8$ to $9 \sim 10^8$ years, the Universe experienced the *reionization period* (92). Gradually, UV light from the first stars ionized the hydrogen atoms into electrons and protons. By the time the Universe reached approximately $\sim 10^9$ years, it had already been almost completely reionized by star formation, and it started to resemble the Universe that we observe today. Around 10^{10} years after the Big Bang we reached equality between matter and dark energy, and we entered what is called the *dark energy era.* This era is marked by another phase of accelerated expansion of the Universe, along with a change in the growth of density

fluctuations that implies the progressive dampening of very large-scale structures.

This was just a glimpse of the history of the Universe – for a complete review, see the References (75, 93).

2.3 Cosmology Toolbox

One of the goals of Cosmology is to transform astronomical observations into information that can describe the evolution and structures that we see in the Universe. In this section, we will summarize the main methods we use in this thesis, following the References (2–4, 41, 76).

2.3.1 Transforming structure formation into structure information

Due to the nature of DM, observations allow us to access only one part of the matter content of the Universe, namely *tracers*: galaxies, quasars and other objects that typically correspond to large concentrations of baryonic matter. Since these tracers reside inside large concentrations of DM, by measuring their positions we can trace the density field of DM, denoted by $\rho(\mathbf{r})$.

In order to understand not only the density but also the fluctuations of matter, we can define the *density contrast* (2–4, 41)

$$\delta(\mathbf{r}) = \frac{\rho(\mathbf{r}) - \bar{\rho}}{\bar{\rho}}, \tag{2.17}$$

where $\bar{\rho}$ represents the average density of DM. This quantity can be initially understood as a Gaussian random field, with higher values indicating a greater concentration of matter in some given region. By definition, its average value is zero and $\delta \in [-1, \infty)$.

The density contrast also allows us to describe the distributions of objects such as tracers in space. The information about the degree of inhomogeneity in the Universe can be better understood by comparing the resulting distribution with a random but completely uniform distribution. In this manner, we can define the correlation between two points separated by a distance r by considering the probability of finding a pair of objects (or particles) in two volume elements dr_1^3 and dr_2^3 around those points, given by

$$dP(\mathbf{r}_1, \mathbf{r}_2) = dr_1^3 \, dr_2^3 \, \rho^2 \left[1 + \xi(r)\right], \tag{2.18}$$

where ρ is their density and $\xi(r) = \langle \delta(\mathbf{r}_1)\delta(\mathbf{r}_2) \rangle$ represents the *two-point correlation function* (2, 3, 41). In the sense above, the correlation function is the excess probability (with respect to the uniform distribution), per unit volume, of finding two objects separated by a distance r.

Simultaneously, another method to convey the same information is through the *power spectrum*, denoted by $P(k)$. It can be defined using the overdensity in Fourier space as

$$\tilde{\delta}(\mathbf{k}) = \int d^3 \, r e^{-i\mathbf{k}\cdot\mathbf{r}} \, \delta(\mathbf{r}), \tag{2.19}$$

such that

$$\langle \tilde{\delta}(\mathbf{k}_1)\, \tilde{\delta}(\mathbf{k}_2) \rangle = (2\pi)^2\, \delta^D(\mathbf{k}_1 - \mathbf{k}_2)\, P(k). \tag{2.20}$$

In the previous equation, $\delta^D(\mathbf{k}_1 - \mathbf{k}_2)$ is the 3D Dirac delta function, and the scale of fluctuations is given by λ, which is related to the wave number $k = 2\pi/\lambda$ (2–4, 94). Therefore, the power spectrum measures the amount of power in the fluctuations at different scales.

By definition, the correlation function and the power spectrum are related through a Fourier transform:

$$P(k) = \int d^3r\, e^{-i\mathbf{k}\cdot\mathbf{r}}\, \xi(r). \tag{2.21}$$

Both quantities are extensively utilized in Cosmology. In this work, we primarily employ the power spectrum as the summary statistic of choice for our analysis (see Chapter 4 and Section 5.3).

2.3.2 Linking observations to theory

One of the most crucial tasks in Cosmology is bridging observations with the theories we develop to interpret them. While we understand that the Universe is primarily composed of DM (as discussed in Chapter 1), this entity is not directly measurable (14). Instead, we observe galaxies, among other things. Galaxies, in turn, "trace" the field of DM δ_m to which they belong through a quantity called *bias*, denoted by b_g, where the sub-index g was used here to indicate that we refer to galaxies (3, 76, 95). This relation can be expressed as follows:

$$\delta_g = b_g\, \delta_m, \tag{2.22}$$

where δ_g represents the density contrast of galaxies. Thus, when $b \neq 1$, signifies that the tracer in question does not perfectly follow the total distribution of matter[2] (2).

Another important aspect regarding galaxies is their discrete nature. We can represent the spatial distribution of discrete objects as a number density of localized point-like particles:

$$n_g(\mathbf{r}) = \sum_{i=1}^{N} \delta^D(\mathbf{r} - \mathbf{r}_i)\ . \tag{2.23}$$

The mean number of galaxies in a volume V is then given by

$$\bar{n}_g = \frac{1}{V} \int_V d^3r\, n_g(\mathbf{r}) = \frac{N_g}{V}\ . \tag{2.24}$$

Therefore, the galaxy density contrast is expressed as

$$\delta_g(\mathbf{r}) = \frac{n_g(\mathbf{r}) - \bar{n}_g}{\bar{n}_g}, \tag{2.25}$$

[2] Note that with the presented definition of bias (and the subsequent one, see Section 2.3.4), we are assuming the approximation of linear and scale-independent bias.

where $n_g(\mathbf{r}) = N_g(\mathbf{r})/\Delta V$. We can model the (random) counts of galaxies in a given volume in terms of the Poisson distribution, and the statistical fluctuations due to this random point process contribute some noise to the power spectrum. With all these ingredients, the distribution of galaxies can be measured using

$$P_g(k) = b_g^2 \, P_m(k) + \frac{1}{\bar{n}_g}, \tag{2.26}$$

where b_g represents the galaxy bias, $P_m(k)$ denotes the DM power spectrum, and $1/\bar{n}_g$ is referred as *shot noise* or *Poisson noise* (2). As mentioned above, this noise arises because we are connecting a discrete distribution (galaxies, following a Poisson distribution) to a continuous distribution (DM, following a Gaussian distribution).

2.3.3 Computing the power spectrum

The power spectrum can be measured directly from the definition given by Equation 2.20. In other words, when we say we are computing the power spectrum, we are employing Fast Fourier Transformations (FFT)[3] of the density contrasts, taking their quadratic modulus, averaging over the values of k in some bin, and removing the shot noise (if we happen to do this in the case of discrete tracers).

However, this simple analysis does not necessarily achieve the lowest possible uncertainties in the power spectra measurements: for that we need more sophisticated estimators of the power spectrum, such as the *FKP estimator* (94). The FKP estimator, developed in 1994 by Hume Feldman, Nick Kaiser, and John Peacock (2, 94), measures the spectra in basically the way we previously described, but with the crucial difference that regions with higher signal-to-noise ratios are upweighted relative to regions with more noise. This is achieved by multipliying the density field by a *weight function* ω_{FKP}, known as the FKP weights, and which can be written as:

$$\omega_{FKP}(\mathbf{r}) = \frac{1}{1 + \bar{n}(\mathbf{r}) P_0(k)} \, , \tag{2.27}$$

where $\bar{n}(\mathbf{r})$ is the mean density and $P_0(k)$ is the power spectra corresponding to the desired optimization region[4]. We make use of the FKP spectra estimation in Section 5.3.

2.3.4 The number density of objects and the halo model

The abundance of astronomical objects can be represented by their *mass function*, or *halo mass function* in the case of DM halos. As we mentioned before, at a basic level the

[3] FFT comprehend an algorithm that compute discrete Fourier transformations. Fourier analysis means the conversion of a signal from its domain (usually time or real space) to its representation in the Fourier space k. Then, a discrete Fourier transformation is obtained decomposing a sequence of values on their components in different frequencies.

[4] It can be odd the fact we need the power spectrum (at certain scale) to compute the power spectrum, as the authors comment in the Reference (94). But, usually, this value is taken as a constant, assuming a value of $P_0 \sim 10^4 \mathrm{Mpc}^3/h$ at $z = 0$ in galaxy surveys (96).

Universe is composed of DM, and the *halo model* (97) provides a semi-analytical description of the behavior of these entities. According to this model, most of the matter in the Universe is contained in DM halos, which form a web of structures composed by *nodes*, *voids*, *walls* and *filaments*.

Halos are gravitationally bound regions of radius R and mass

$$M = \frac{4\pi}{3}\rho_m(z)R^3, \tag{2.28}$$

where $\rho_m(z) = \rho_{crit}\Omega_m(1+z)^3$ and ρ_{crit} is the critical matter density, $\Omega_m = \frac{\rho_m}{\rho_{crit}}$. Here R represents the radius of the original region (at $z \sim 10^3$) that eventually collapsed to form the halo.

The mass function describes the probability of having a number of halos in a certain *redshift* z for a specific mass interval $[\ln M, \ln M + d\ln M]$. It is given by

$$\frac{dn\,(z,M)}{d\ln M} = f(\sigma)\,\frac{\bar{\rho}_{m0}}{M}\,\frac{d\ln\sigma^{-1}}{d\ln M}, \tag{2.29}$$

where n is the halo number counts per unit volume, $f(\sigma)$ is the multiplicity function, $\bar{\rho}_m$ is the matter density, and σ is the variance of the density fluctuations smoothed on the sphere of radius R which corresponds to the mass M. The multiplicity function can be analytically (or semi-analytically) computed using the spherical collapse model (98, 99), ellipsoidal collapse (100), or it can be inferred from numerical fits using N-body simulations (101, 102). The general behavior of this equation is such that the number of halos decreases with mass, in agreement of the general idea of a hierarchical scenario of structure formation.

On the other hand, the position and relative abundance of the halos are also related to their linear bias b, which can be expressed as the ratio of the halo power spectra $P_{halos}(k)$ to the linear[5] DM spectrum $P_{lin}(k)$ (76, 103)

$$b^2(k) = \frac{P_{halos}(k)}{P_{lin}(k)}. \tag{2.30}$$

This quantity can be derived analytically, starting from the mass function (98), or it can be obtained from numerical fits based on N-body simulations (102, 103).

We will compute and then compare the halo mass functions, as well the halo bias, in Section 2.5.1.8 and in Chapter 4, where we also show that more massive halos are (i) less abundant and (ii) more clustered (i.e., they have higher bias).

2.3.5 Bayesian statistics, parameter inference, and cosmological covariance matrices

Bayesian statistics and its associated methods start with the premise that statements expressed through probabilities are not limited to data alone, but should extend to our choices

[5] We will present the idea behind this nomenclature (linear spectra) while presenting the linear perturbation in Section 2.4.

for different models and their parameters as well (41). Cosmology has benefited immensely from these ideas, given the fundamental fact that we have only one observable Universe. The Bayesian toolbox has become an essential part of our model-testing and statistical inference pipelines for constraining model parameters, in particular in the area of large-scale structures (104).

The starting point is of course Bayes theorem, which can be written as (2, 41):

$$p(\boldsymbol{\theta}|\mathbf{D}, M) = \frac{p(\mathbf{D}|\boldsymbol{\theta}, M)p(\boldsymbol{\theta}|M)}{p(\mathbf{D}|M)}. \tag{2.31}$$

It states that we can compute the probability distribution of parameters $p(\boldsymbol{\theta}|\mathbf{D}, M)$ (or *posterior*), given the parameters $\boldsymbol{\theta}$ conditioned on the data D, following a model M. This is a function of three components: $p(\mathbf{D}|\boldsymbol{\theta}, M)$, the probability of having the data D conditioned on the parameters $\boldsymbol{\theta}$, following the model M (known as the *likelihood*); $p(\boldsymbol{\theta}|M)$, the probability of having the parameters $\boldsymbol{\theta}$ conditioned on the model M (known as *prior*); and $p(\mathbf{D}|M)$, the probability of having the data $\mathbf{D}$ conditioned on the model M (i.e., the *marginalization*).

This theorem is used to obtain the best set of parameters that can represent a data set, given a model. In other words, the *likelihood* is maximized by exploring an interval of theoretical parameters, allowing us to find the set that best fits the data (105).

When the search requires a large number k of parameters for a model $M(\boldsymbol{\theta})$, with a parameter array given by $\boldsymbol{\theta}$, *Markov Chain Monte Carlo* (MCMC) methods are used (41). These methods are based on algorithms that compute the probability distribution based on Markov chains – i.e., points in parameter space whose likelihoods are known. These chains are constructed by means of an interactive process where future states (the points in the chains) depend only on the present state, not on the past.

The basic idea is to express the *likelihood* L as a distribution χ^2, as follows (2, 3)

$$L \propto e^{-\chi^2/2}. \tag{2.32}$$

Here, χ^2 accounts for the difference between the data and the model, defined as the trace:

$$\chi^2 = \left[D - M(\boldsymbol{\theta})\right]^T Cov^{-1} \left[D - M(\boldsymbol{\theta})\right]. \tag{2.33}$$

In this equation, D represents the data vector, $M(\boldsymbol{\theta})$ denotes the model vector, and Cov is the *covariance matrix* of the data vectors, whose inverse is denoted by Cov^{-1}. The MCMC algorithms search for stationary distributions of the parameters by minimizing this parameter, thereby finding the best model description for the given data set.

2.3.5.1 Covariance matrices

The concepts of *covariance* and *correlation* can be applied to any variables of interest, in order to measure their inter-dependency (41). Here, we will define them in terms of the power spectrum $P(k)$, as it is the summary statistic used in Chapter 4.

Computing the covariance for the power spectra means measuring the dependence between each *bin* k in Fourier space, which is here represented by indices i and j. Therefore, the *sample covariance matrix*, denoted by Cov_{ij}, is a square positive-definite matrix that measures any linear relationships that may be present between the spectra at different bins, and can be written as:

$$Cov_{ij}^{(N)}[P(k_i), P(k_j)] = \frac{1}{(N-1)} \sum_{l=1}^{N} \left[P(k_i)_l - \bar{P}(k_i)\right] \left[P(k_j)_l - \bar{P}(k_j)\right], \tag{2.34}$$

where N is the number of spectra in the data vector (the sample size), $P(k_i)_l$ is the value of the l-th spectra for the i-th bin, and $\bar{P}(k_i)$ is the mean power spectrum. It is worth noting that the covariance matrix contains elements with a wide range of values, which often makes comparisons and visualization challenging. To address this, we define the *correlation matrix*

$$Corr_{ij} = \frac{Cov_{ij}}{norm_{ij}}, \qquad norm_{ij} = \sqrt{Cov_{ii}\, Cov_{jj}}, \tag{2.35}$$

where $norm_{ij}$ is a normalization factor for bin pairs $\{k_i, k_j\}$. The correlation matrix can be seen as an adimensional, renormalized version of the covariance matrix whose diagonal values are unity. In fact, we can normalize any covariance matrix through the $norm_{ij}$ factor, and we can also refer to the result as the *normalized covariance matrix*.

One of the main issues that often affect sample covariances is the limited size of the samples: the number of data points available may be small for a number of reasons. In particular, with only a small number of spectra we may fail to capture the true correlations between different values in the different bins. Therefore, we must be able to collect a large amount of data in order to obtain well-estimated covariance matrices, which, in turn, will lead to more accurate parameter estimations. We will address this problem in Chapter 4.

2.4 Cosmology in the linear regime

The Universe behaves as a homogeneous and isotropic fluid at large scales (distances $\gtrsim 100$ Mpc/h), as evidenced by observations such as the CMB (13, 14). However, on smaller scales, we observe galaxies and clusters forming what is known as the *cosmic web*, indicating that the Universe is not locally homogeneous. Understanding the evolution of deviations from the background is crucial for comprehending the formation of galaxies and structures which are visible today. Ultimately, these objects originate from quantum fluctuations during inflation, emerging as small perturbations on top of a homogeneous and isotropic background (75).

Initially, these fluctuations were very small, allowing them to be treated as *linear perturbations*. They grow over time, eventually shaping the cosmological structures observed today. Even for current observations, for scales larger than ~ 100 Mpc/h, the observed inhomogeneities are very small (when density perturbations are very small, i.e., $\delta \ll 1$). Thus, on large scales the Universe can be described using *linear perturbation theory*, which includes

perturbations in both matter (dark and baryonic) and radiation. In this section we will provide a brief overview of this model – for a complete discussion see Reference (3).

We can start the overview of linear perturbation theory by considering a matter fluid with small perturbations. For example, we can write the density field as

$$\rho\left(\mathbf{x},t\right)=\bar{\rho}\left(t\right)+\delta\rho\left(\mathbf{x},t\right), \tag{2.36}$$

where $\bar{\rho}\left(\mathbf{x},t\right)$ represents the background quantity (see Equations 2.13, 2.14, 2.15 for each matter component), and $\delta\rho\left(\mathbf{x},t\right)$ represents the density perturbation. Our goal here is to understand how an ensemble of density fluctuations evolve with time. This can be achieved by considering *perturbations* to the energy density and pressure off all matter components and using the *Boltzmann equations* together with the FLRW metric that describes this nearly homogeneous and isotropic spacetime.

Due to the density and pressure perturbations the metric itself must also present deviations from perfect homogeneity and isotropy. The linear density perturbations in the FLRW metric are taken into account by considering the so-called scalar metric perturbations, given by

$$ds^{2}=-\left[1+2\Psi\left(\mathbf{x},t\right)\right]dt^{2}+a^{2}(t)\left[1+2\Phi\left(\mathbf{x},t\right)\right]dr^{2}, \tag{2.37}$$

where $\Psi\left(\mathbf{x},t\right)$ represents the Newtonian gravitational potential, $\Phi\left(\mathbf{x},t\right)$ represents the curvature potential, and dr^{2} represents the spatial component of the metric.

While this expression may appear simple, it is important to note that it is connected to all the matter components in the Universe (due to the energy-momentum tensor in the Einstein's equations – see Section 2.1). The challenge arises from the way these components interact with each other. For instance, we have to account for Compton scattering between photons and free electrons, as well as Coulomb scattering between electrons and protons (3). The Boltzmann equation can handle these interactions along with all the perturbations in the energy density and pressure. Schematically, the Boltzmann equation tells us how the distribution function f_i of some matter component i evolves with time:

$$\frac{df_{i}}{dt}=C\left(f_{i}\right). \tag{2.38}$$

In this expression, C is the collision term, which carries information about the interactions and may take a complicated form depending on the type of interactions.

For DM, the variables of interest are the density contrast $\delta(\mathbf{x},t)=\delta\rho(\mathbf{x},t)/\bar{\rho}(t)$ and the velocity $\mathbf{v}(\mathbf{x},t)$. The equivalent perturbation for baryons is represented by $\delta_b(\mathbf{x},t)$ and $\mathbf{v}_b(\mathbf{x},t)$. In the case of photons, the perturbation in the Bose-Einstein equilibrium distribution function is characterized by $\Theta(k,\mu,\eta)=\delta T/T$, which may depend both on the spatial position through the Fourier wave vector $\mathbf{k}$, as well as the photon momenta (or direction) $\mathbf{p}$ through its projection $\mu\equiv\hat{p}\cdot\hat{k}$. The temperature fluctuation can also depend on time, which is here

expressed by conformal time η where $d\eta = dt/a(t)$. For neutrinos, the distribution function is given in terms of their temperature as well: $\mathcal{N}(k, \mu, \eta) = \delta T_\nu / T_\nu$. Regarding the polarization fraction of the photon field, we denote it as Θ_P.

Notice that the momentum dependence of the photon field (both in terms of its temperature and polarization) is included in δT through its projection μ. In order to include the degree of anisotropy of the photon field it is useful to split these quantities into *multipoles* of the photon temperature distribution, expanded in terms of Legendre polynomials P_ℓ:

$$\Theta_\ell = \frac{i^\ell}{2} \int_{-1}^{1} d\mu \; \Theta(t, k, \mu) \; P_\ell(\mu). \tag{2.39}$$

Let's consider the main channel for interaction between baryonic matter and radiation – Compton scattering. The rate of interactions depends not only on the availability of free electrons, expressed by their number density n_e, but also on the baryon velocity $\mathbf{v}_b$, and of course the Thompson cross-section σ_T. The probability that photons will interact with free electrons via Compton scattering is usually encapsulated by the optical depth:

$$\tau \equiv \int_{\eta}^{\eta_0} d\eta \; n_e \; \sigma_T \; a. \tag{2.40}$$

In other words, the probability per unit (conformal) time that a photon is Compton-scattered by a free electron is $d\tau/d\eta = n_e \, \sigma_T \, a$.

The Boltzmann equations connecting all the matter components, the photon temperature (Θ) and polarization (Θ_P), dark (δ) and baryonic (δ_b) matter, and neutrinos ($\mathcal{N}$), are given by (3):

$$\dot{\Theta} + ik\mu\Theta = -\dot{\Phi} - ik\mu\Psi - \dot{\tau}\left[\Theta_0 - \Theta + \mu v_b - \frac{1}{2}P_2(\mu)\Pi\right], \tag{2.41}$$

$$\dot{\Theta}_P + ik\mu\Theta_P = \dot{\tau}\left\{-\Theta_P + \frac{1}{2}\left[1 - P_2(\mu)\right]\Pi\right\}, \tag{2.42}$$

$$\Pi = \Theta_2 + \Theta_{P_2} + \Theta_{P_0}, \tag{2.43}$$

$$\dot{\delta} + ikv = -3\dot{\Phi}, \tag{2.44}$$

$$\dot{v} + \frac{\dot{a}}{a}v = -ik\Psi, \tag{2.45}$$

$$\dot{\delta}_b + ikv_b = -3\dot{\Phi}, \tag{2.46}$$

$$\dot{v}_b + \frac{\dot{a}}{a}v_b = -ik\Psi + \frac{\dot{\tau}}{R}\left(v_b + 3i\Theta_1\right), \tag{2.47}$$

$$\dot{\mathcal{N}} + ik\mu\mathcal{N} = -\dot{\Phi} - ik\mu\Psi. \tag{2.48}$$

Notice that Equation 2.47 takes into account the ratio of photon to baryon densities, defined by:

$$\frac{1}{R} \equiv \frac{4\rho_\gamma}{3\rho_b}. \tag{2.49}$$

As for the gravitational dynamics, it is found by using the perturbed FLRW metric in Einstein's Field Equations, expanding to first order in the perturbations, and then coupling that to the (perturbed) energy-momentum tensor. We end up with equations for the potentials given by:

$$k^2\Phi + 3\frac{\dot{a}}{a}\left(\dot{\Phi} - \Psi\frac{\dot{a}}{a}\right) = 4\pi G a^2\left(\rho_m\delta_m + 4\rho_r\Theta_{r,0}\right) \tag{2.50}$$

$$k^2\left(\Phi + \Psi\right) = 32\pi G a^2\rho_r\Theta_{r,2}\,. \tag{2.51}$$

Here, the subscript m includes all matter (baryons and dark matter), and the subscript r denotes the entire radiation content (neutrinos ν and photons γ):

$$\rho_m\delta_m = \rho\delta + \rho_b\delta_b \tag{2.52}$$

$$\rho_r\Theta_{r,0} = \rho_\gamma\Theta_0 + \rho_\nu\mathcal{N}_0 \tag{2.53}$$

$$\rho_r\Theta_{r,1} = \rho_\gamma\Theta_1 + \rho_\nu\mathcal{N}_1. \tag{2.54}$$

Before attempting to solve these coupled equations we must settle on the *initial conditions* (ICs), which are usually assumed to be given in terms of a primordial spectrum of fluctuations that results from some *inflationary model* (3, 84, 85, 106). Although explaining inflation is outside the scope of the present thesis, it remains the most plausible explanation for the mechanism that underlies the near-uniformity of the density and temperature of the Universe, as well as the origin of the perturbations.

The complete set of Equations 2.41-2.50 can be solved analytically only for certain scales and up to certain moments in time (3). In order to simplify this analysis we can schematically write the solution for the potential Φ as:

$$\Phi(\mathbf{k}, a) = \Phi_p(\mathbf{k})\;T(k)\;D(a), \tag{2.55}$$

where $\Phi_p(\mathbf{k})$ is the primordial value of the potential, $T(k)$ is a transfer function, and $D(a)$ is the growth factor. The *transfer function* describes the relative growth (or damping) of different modes from one instant in time to another. In particular, in this context it will express the evolution of perturbations through the epochs of horizon crossing and radiation/matter transition, a period which is called the *transfer function regime*[6]. It is useful to define the transfer function in such a way that it is equal to unity on the largest scales, which leads us to:

$$T(k) \equiv \frac{\Phi(k, a_{late})}{\Phi_{Large-Scale}(k, a_{early})}, \tag{2.56}$$

where a_{late} denotes an epoch well after the transfer function regime, and $\Phi_{Large-Scale}(k, a_{early})$ is the primordial value of Φ. This quantity can be obtained using numerical fits such as Bardeen, Bond, Kaiser and Szalay (BBKS) (107) or that by Eisenstein and Hu (108).

[6] It represents the period just after inflation, throughout the epoch of horizon crossing, to the epoch of matter-radiation equality. *Horizon crossing* is when the wavelength of a mode becomes smaller than the Hubble radius, $k < aH$.

On the other hand, the *growth function* describes the wavelength-independent growth of perturbations (which is especially useful at late times). It is also named as D_1 when defined for $a > a_{late}$:

$$D_1(a) \equiv a \, \frac{\Phi(a)}{\Phi(a_{late})}. \tag{2.57}$$

In fact, in a matter-dominated Universe, $D_1(a) = a$. With these conventions, the potential is rewritten as

$$\Phi(\mathbf{k}, a) = \frac{9}{10} \, \Phi_p(\mathbf{k}) \, T(k) \, \frac{D_1(a)}{a}, \tag{2.58}$$

where the factor $9/10$ takes into account the transition from the radiation to the matter era, which boosts the potential on large scales by a factor $10/9$.

It should be stressed now that the potential Φ cannot be directly measured. We employ the power spectra of the density fluctuations to measure the matter distribution, and then compare the observations to the linear theory and the coupled predictions for the density and Φ. In order to draw this correspondence we can relate the matter density constrast δ to the potential Φ using the Poisson equation:

$$\Phi = \frac{4\pi G \rho_m a^2 \delta}{k^2}. \tag{2.59}$$

After some manipulation, we get that, at late times:

$$\delta(\mathbf{k}, a) = \frac{3}{5} \, \frac{k^2}{\Omega_m H_0^2} \, \Phi_P(\mathbf{k}) \, T(k) \, D_1(a) \, . \tag{2.60}$$

Assuming that $\Phi_P(\mathbf{k})$ is drawn from a Gaussian distribution such that $P_\Phi = (50\pi^2/9k^3) \, (k/H_0)^{n-1} \, \delta_H \, [\Omega_m / D_1(a=1)]^2$, the power spectrum of the matter perturbations is given by

$$P(k, a) = 2\pi^2 \, \delta_H^2 \, \frac{k^n}{H_0^{n+3}} \, T^2(k) \left[\frac{D_1(a)}{D_1(a=1)} \right]^2 , \tag{2.61}$$

where δ_H is the density constrast at the horizon crossing.

Indeed, the description of the power spectrum provided by Equation 2.61 is not straightforward, but it leads to the well-known result for the scaling of the power spectrum:

$$P(k) \propto k^n \, . \tag{2.62}$$

The scale-invariant Harrison-Zeldovich spectrum corresponds to case $n = 1$ (109, 110). Another interesting point is that, by considering the BBKS transfer function (107), we get a fair approximation of the shape of the spectra as

$$P(k) \propto k T^2(k) = \begin{cases} k, & k < k_{eq} \\ \frac{\ln^2 k}{k^3}, & k > k_{eq} \end{cases} , \tag{2.63}$$

where $k_{eq} = a_{eq} H(a_{eq})$ represents the horizon scale at matter-radiation equality (see Section 2.1).

To accurately predict the linear evolution and exact shape of the power spectrum for different cosmological models, numerical codes are required to solve the entire set of perturbation equations, such as CAMB (Code for Anisotropies in the Microwave Background) (111). With the linear spectrum at hand, we can even compute the halo abundance and the linear bias (see Section 2.3.4).

2.5 Nonlinear Cosmology

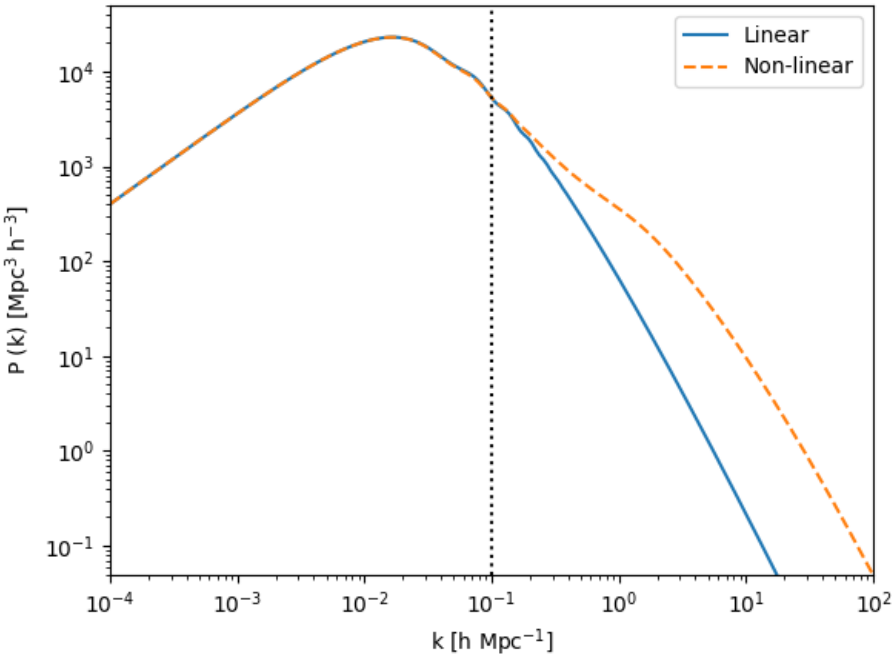

Figure 4 – **Linear and nonlinear power spectrum comparison.** The spectra are presented for today's values ($z = 0$), and were obtained with CAMB for the Quijote fiducial Cosmology. The impact of nonlinearities is evident starting from $k \gtrsim 0.1\,h/\mathrm{Mpc}$.

Linear perturbation theory is all we need for studying large-scales ($k \lesssim 0.1\,h/\mathrm{Mpc}$). However, by the time fluctuations reach order unity, $\delta \sim 1$, linear perturbation theory has already broken down. This breakdown becomes evident in the power spectrum, depicted in Figure 4 – here we adopt the Quijote fiducial Cosmology: $\Omega_m = 0.3175$, $\Omega_b = 0.049$, $h = 0.6711$, $n_s = 0.9624$, $\sigma_8 = 0.834$, $M_\nu = 0.0eV$, and $\omega = -1$ (112), also in accordance with Planck (14). The power spectra shown in the figure were computed using CAMB (111), where we also highlight the scale $k = 0.1h/\mathrm{Mpc}$ with a dotted vertical line, which marks the onset of nonlinearities. The two presented spectra start to diverge approximately at this scale, with the power of k modes enhanced for small scales in the nonlinear case.

There are several branches that attempt to address the limitations of linear theory. One approach involves considering nonlinear terms in the perturbation theory, similar to what we have seen in the previous section, referred to as N-order perturbation theory ($N-$PT) (113, 114). Another approach involves non-perturbative methods, such as the *halo model*, which

we briefly discussed in Section 2.3.4, but only in the context of linear theory. In the context of the halo model, the linear power spectrum is expressed by the 2-halo term, while the "shoulder" at small scales arises from the 1-halo term (115). Alternatively, simpler approaches like the Zeldovich approximation (110, 116) are also used. Each of these methods has its advantages and disadvantages. For example, while some may require higher-order perturbation terms or additional parameters (117), others (like halo model) may not guarantee exact recovery of perturbation theory results on large scales (118).

To predict clustering on small scales, the main approach is through *numerical simulations*. These simulations employ high-resolution, multi-scale schemes and are routinely used on massively parallel computers to increase their size and complexity, aiming to better describe the Universe. Given the complexity of the Universe and its various components, these numerical solutions can be classified into two main categories: N-body (or DM-only) simulations and hydrodynamical (or DM plus baryons) simulations. N-body simulations involve solely DM, with gravity being the only force acting on the particles. On the other hand, hydrodynamical simulations incorporate both DM and baryonic matter (e.g., gas), allowing for the inclusion of phenomena such as feedback from supernova explosions and supermassive black holes (BH), magnetic effects, and more. Since these simulations form the core of the data sets used in the present thesis, this section will briefly cover both types. Additionally, we will briefly discuss some approximate methods developed to yield results similar to these simulations, but in a faster manner.

2.5.1 N-body simulations

N-body (or DM-only) simulations are numerical solutions of a very high number (N) of DM particles interacting gravitationally within a finite volume, and evolving over a long period of time (or a large range of redshifts) (119, 120). Essentially, they provide an alternative path to solutions of the collisionless Boltzmann equations coupled with Poisson's equation. Examples of such simulations include the Millennium (121), Dark Sky (122), and Bolshoi (123) simulations, which are widely used by the scientific community to study large-scale structures and the behavior of DM on large volumes. Another notable example is the Quijote project (112), which consists of a collection of $43,100$ full N-body simulations designed to provide an extensive data set of cosmological simulations for ML applications. These simulations employ DM-only particles and utilize the TreePM code GADGET-III, which is the third generation of the well-known GADGET-II algorithm (124). These simulations are instrumental in validating the robustness of the work presented in Chapter 4.

In this Section we will present one of the techniques employed to solve the problem of simulating cosmological structures: the *particle mesh* (PM) algorithm. Other methods, such as particle-particle schemes or hybrid schemes, also exist, each with its own advantages and disadvantages, which are detailed in References (125, 126).

The PM method is a relatively simple approach that I used myself when first delving into Cosmology. I coded my own version of the method, largely following the principles outlined in Reference (127). PM codes utilize a mesh to represent density and potential fields, with the resolution of the simulation limited by the size of this mesh. Despite its simplicity, the PM method offers several advantages. It is fast, requiring fewer operations per particle per time step, compared to other methods. Additionally, PM simulations can handle very large numbers of particles efficiently.

Numerical N-body algorithms allow the study of nonlinear gravitational evolution of complex particle systems. These simulations model the time evolution of a given system by determining and tracking the trajectories of particles, taking into account their mutual gravitational interactions (128). Thus, a PM code solves both the Poisson equation,

$$\nabla^2\Phi = 4\pi G\,\Omega_{m,0}\,\rho_{crit}\,a^{-1}\delta\,, \tag{2.64}$$

as well as the equations of motion of the particles,

$$\frac{d\mathbf{x}}{da} = \frac{\mathbf{p}}{\dot{a}\,a^2} \tag{2.65}$$

$$\frac{d\mathbf{p}}{da} = -\frac{\nabla\Phi}{\dot{a}}\,. \tag{2.66}$$

These equations are written in terms of comoving coordinates, i.e., $\mathbf{x} = \mathbf{r}/a$, where $\mathbf{r}$ represents the proper particle's fluid position, and $\mathbf{p} = a\mathbf{v} = a^2\dot{\mathbf{x}}$ denotes the particle momenta, where

$$\mathbf{v} = \mathbf{u} - H\mathbf{r} = a\dot{\mathbf{x}} \tag{2.67}$$

is the peculiar velocity, and $\mathbf{u}$ is the proper velocity (including the Hubble flow).

It is convenient to define code variables, i.e., dimensionless variables, that we will denote with tildes according to:

$$\tilde{\mathbf{x}} \equiv \frac{\mathbf{x}}{r_0} = \frac{\mathbf{r}}{ar_0}, \tag{2.68}$$

$$\tilde{\mathbf{p}} \equiv \frac{\mathbf{p}}{v_0} = \frac{a\mathbf{v}}{v_0}, \tag{2.69}$$

$$\tilde{\Phi} \equiv \frac{\Phi}{\phi_0}, \tag{2.70}$$

$$\tilde{\rho} \equiv a^3\frac{\rho}{\rho_0}. \tag{2.71}$$

The quantities with subscript zero correspond to physical variables responsible for removing the units from the code variables and are defined as

$$r_0 \equiv \frac{L_{BOX}}{N_g}, \tag{2.72}$$

$$t_0 \equiv \frac{r_0}{t_0}, \tag{2.73}$$

$$\rho_0 \equiv \rho_{crit}\Omega_{m,0}, \tag{2.74}$$

$$\Phi_0 \equiv \frac{r_0^2}{t_0^2} = v_0^2, \tag{2.75}$$

where L_{BOX} is the box size, measured in Mpc$/h$, N_g is the number of grid cells in each direction, and $N_g^T = N_g^3$ is the total number of grid cells. In dimensionless variables, Equations 2.64-2.66 can be rewritten as

$$\tilde{\nabla}^2\tilde{\Phi} = \frac{3}{2}\frac{\Omega_{m,0}}{a}\tilde{\delta}, \tag{2.76}$$

$$\frac{d\tilde{\mathbf{x}}}{da} = f(a)\frac{\tilde{\mathbf{p}}}{a^2}, \tag{2.77}$$

$$\frac{d\tilde{\mathbf{p}}}{da} = -f(a)\tilde{\nabla}\tilde{\Phi}, \tag{2.78}$$

where $\tilde{\delta} = \tilde{\rho} - 1$ and $f(a) \equiv H_0/\dot{a}$.

The main idea of the PM code is to solve the Equations 2.76-2.78 in three main steps:

1. Solve the Poisson Equation 2.76 using the density field, estimated with current particle positions;
2. Advance momenta $\tilde{\mathbf{p}}$, using the potential computed in the first step;
3. Update particle positions $\tilde{\mathbf{x}}$, using the advanced momenta.

2.5.1.1 Implementation stage: solving the equations

The PM method exploits the fact that the Poisson equation for gravitational potential (see Equation 2.76) can be found in real space by convolving the density contrast with the Green's function

$$\tilde{\phi}(\tilde{\mathbf{x}}) = \int d^3\tilde{x}'\, G(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')\, \tilde{\delta}(\tilde{\mathbf{x}}'). \tag{2.79}$$

The choice of the particular Green's function G is driven by the fact that we use periodic boundary conditions (PBC) – see below, Equation 2.85. In Fourier space, the convolution is then replaced by a simple multiplication:

$$\tilde{\phi}(\tilde{\mathbf{x}}) = G(\mathbf{k})\, \tilde{\delta}(\mathbf{k}). \tag{2.80}$$

To obtain the density contrast $\tilde{\delta}(\mathbf{k})$ in Fourier space, first it is necessary to obtain $\tilde{\delta}(\tilde{\mathbf{x}})$ in real space, which arises from the density in real space $\tilde{\rho}(\tilde{\mathbf{x}})$ (129).

2.5.1.2 The density field

In PM algorithms, particles are assumed to have a certain size, mass, shape, and internal density. This determines the interpolation scheme used to assign densities to grid cells (127,129). A common choice is the *Cloud In Cell* (CIC) method, where particles are represented as cubes (in 3D) of uniform density and of one grid cell size.

The algorithm described above is relatively computationally cheap, accurate, and is commonly used in PM codes. In this method, the shape function of a particle in 1 dimension is

defined as

$$S(\tilde{x}) = \frac{1}{\Delta\tilde{x}} \begin{cases} 1, & |\tilde{x}| < \Delta\tilde{x}/2 \\ 0, & \text{otherwise} \end{cases}, \tag{2.81}$$

for a cell size of $\Delta\tilde{x}$. Then, the mass fraction of particle at $\tilde{x}_p$, assigned to a cell at $\tilde{x}_{ijk}$, is the shape function averaged over this cell:

$$W(\tilde{x}_p - \tilde{x}_{ijk}) = \int_{\tilde{x}_{ijk}-\Delta\tilde{x}/2}^{\tilde{x}_{ijk}+\Delta\tilde{x}/2} d\tilde{x}'\, S(\tilde{x}_p - \tilde{x}'). \tag{2.82}$$

In 3 dimensions this process generalizes to

$$W(\tilde{\mathbf{x}}_p - \tilde{\mathbf{x}}_{ijk}) = W(\tilde{x}_p - \tilde{x}_{ijk})\, W(\tilde{y}_p - \tilde{y}_{ijk})\, W(\tilde{z}_p - \tilde{z}_{ijk})\,, \tag{2.83}$$

such that the density $\tilde{\rho}_{ijk}$ in the corresponding cell is given by

$$\tilde{\rho}_{ijk} = \sum_{p=1}^{N_p} \tilde{m}_p W(\tilde{\mathbf{x}}_p - \tilde{\mathbf{x}}_{ijk}), \tag{2.84}$$

where $N_p^T = N_p^3$ is the total number of particles, N_p is the number of particles "on each direction", and m_p is the particle mass[7]. In practice, this is achieved by looping over particles and assigning their density to neighboring cells, rather than summing over all particles for each cell individually.

2.5.1.3 The density contrast field and its Fourier transform

With the grid densities $\tilde{\rho}_{i,j,k}(\tilde{\mathbf{x}})$ on hand, the next step is to obtain the grid density contrasts $\tilde{\delta}_{i,j,k}(\tilde{\mathbf{x}})$ and convert them to Fourier space (127). This transformation is typically accomplished using FFT algorithms, which efficiently compute the discrete Fourier Transform and its inverse. By applying the FFT to the grid density contrasts, we obtain the them in Fourier space, denoted as $\tilde{\delta}_{i,j,k}(\tilde{\mathbf{k}})$.

2.5.1.4 The gravitational potential

According to Equation 2.80, now we only need the Green function to obtain the gravitational field $\tilde{\Phi}(\mathbf{k})$. The Green function, derived in detail in Appendix A, is given by

$$G(\mathbf{k}) = -\frac{3\Omega_{m,0}}{8a}\left[\sin^2\left(\frac{k_x}{2}\right) + \sin^2\left(\frac{k_y}{2}\right) + \sin^2\left(\frac{k_z}{2}\right)\right]^{-1}, \tag{2.85}$$

where

$$k_x = \frac{2\pi l}{L_{BOX}},\;\; k_y = \frac{2\pi m}{L_{BOX}}, k_z = \frac{2\pi n}{L_{BOX}},\;\; \text{for the components } (l, m, n). \tag{2.86}$$

These equations are in code units, hence $L_{BOX} = N_g$. Then, the gravitational potential is solved by transforming the result back to real space to obtain $\tilde{\Phi}(\tilde{\mathbf{x}})$ discretized at cell centers. Note that, when using these gravitational potentials, there is an artifact, a singularity at $l = m = n = 0$, which is avoided by setting $\tilde{\Phi}_{000} = 0$.

[7] Note that the particle mass $\tilde{m}_p$ can be computed in code units as $\tilde{m}_p = N_g^T/N_p^T$, using $\tilde{\bar{\rho}} = 1$. Thus, in the comoving units, the particles mass is $m_p = \tilde{m}_p r_0^3 \rho_0$.

2.5.1.5 The acceleration

After obtaining the gravitational field in real space $\tilde{\Phi}(\tilde{\mathbf{x}})$, discretized at cell centers, it is time to obtain the acceleration at each grid point (127). This is simply given by

$$\tilde{\mathbf{a}}(\tilde{\mathbf{x}_\mathbf{i}}) = -\tilde{\nabla}\tilde{\Phi}(\tilde{\mathbf{x}_\mathbf{i}})\,. \tag{2.87}$$

This step precedes the updating of the particles' positions and momenta, because it requires the accelerations at each particle's position. Thus, to obtain the accelerations at the particle positions $\tilde{\mathbf{g}}^p$, we interpolate the acceleration at grid points $\tilde{\mathbf{a}}(\tilde{\mathbf{x}_\mathbf{i}})$ onto the particle positions $\tilde{\mathbf{x}}_j^p$, using the CIC interpolation. During the density assignment, for a given particle, the acceleration at each point is interpolated from the cells to which the particle contributed to the density.

2.5.1.6 Updating particles positions and momenta

We arrive now at the final stage of the PM method: updating particle positions and momenta. This is achieved using *leapfrog integration* (127,128), which is a numerical method for integrating differential equations in a dynamical system. Leapfrog integration updates positions and velocities (or momenta) at interleaved time points (or scale factor points), staggered in such a way that they "leapfrog" over each other.

Thus, using the leapfrog integration, we have updated momenta and positions as

$$\tilde{\mathbf{p}}_{n+1/2} = \tilde{\mathbf{p}}_{n-1/2} + f(a_n)\tilde{\mathbf{g}}_n \Delta a \tag{2.88}$$

$$\tilde{\mathbf{x}}_{n+1} = \tilde{\mathbf{p}}_n + a_{n+1/2}^{-2} f(a_{n+1/2})\tilde{\mathbf{p}}_{n+1/2}\Delta a, \tag{2.89}$$

where n represents the "time" step, $f(a_n)$ is computed at a_n, Δa is the step in the scalar factor,

$$a_n = a_i + n\Delta a \tag{2.90}$$

is the evolution in the scale factor according to the "time" steps, and $a_{n+1/2} = a_n + \Delta a/2$. In the present case, we always update the momentum first, in a half time step before update the positions.

Therefore, the Particle Mesh (PM) method involves repeating a series of steps for each time step of the simulation. The main scheme of the PM method typically consists of the following three blocks, which are repeated iteratively:

1. **Find density on the mesh using the Cloud-In-Cell (CIC) technique.** This step involves assigning the density of particles to grid cells using the CIC interpolation scheme.

2. **Solve the Poisson equation using two 3-dimensional Fast Fourier Transforms (FFTs).** After obtaining the density distribution on the mesh, the Poisson equation is solved in Fourier space using FFTs to calculate the gravitational potential.

3. **Advance momenta and positions of the particles.** Finally, the momenta and positions of the particles are updated using leapfrog integration based on the calculated gravitational potential.

These blocks are repeated for each time step of the simulation to evolve the system over time.

The simulations were conducted using the following cosmological parameters: $H_0 = 73.0$ km s^{-1} Mpc^{-1}, $\Omega_m = 0.311051$, and $\Omega_\Lambda = 0.68887$. We ran four simulations, which explains the error bars seen in the final products, as presented in the subsequent sections on the realizations over these four boxes. We simulated a box of side $L_{BOX} = 128$ Mpc$/h$, with 128^3 particles and 256^3 cells. The simulations started from $a = 0.02$ or $z = 49$, up to $a = 1.0$ or $z = 0$. The ICs were set using the *Multi Scale Initial Conditions* (MUSIC) code (130). To define the initial spectra using MUSIC we utilized CAMB (111).

2.5.1.7 N-body power spectrum

The primary outcome obtained from the N-body simulation was the *power spectrum*, denoted as $P(k)$, at various stages and configurations of the simulation. This quantity was estimated using Equation 2.20, following the prescription presented in Section 2.3.1. Specifically, we computed the power spectrum based on FFTs performed on the density contrasts. We then calculated the quadratic modulus of these transformations and averaged the results over the k values:

$$P(k) = \langle|\tilde{\delta}(k)|^2\rangle = \frac{1}{N_k}\sum_{i=1}^{N_k}|\tilde{\delta}(k_i)|^2, \tag{2.91}$$

where i represents the *bin* index of k, i.e., k_i lies within the interval $[k, k+\Delta k]$, $k = \sqrt{k_x^2 + k_y^2 + k_z^2}$, and N_k represents the number of points where k_i falls within the respective *bin*.

The power spectrum was computed for two distinct epochs: $a = 0.02$ and $a = 1.0$, to capture the evolution of the simulation. The results are illustrated in Figure 5, where a comparison is made with the linear and nonlinear spectra from CAMB. Error bars in the plots represent the standard deviation of the obtained spectra for four realizations of the simulation conducted under the fiducial Cosmology. Additionally, $\langle std\rangle$ denotes the average of these standard deviation values. The vertical lines denote the confidence interval of k values, computed as

$$k_{min} = \frac{2\pi}{L_{BOX}} \quad \text{and} \quad k_{max} = \frac{\pi N_p}{2L_{BOX}}, \tag{2.92}$$

where L_{BOX} is the size of the box of the simulation, k_{min} is the minimum value of k (also related to the simulation *resolution*), and k_{max} is the maximum value of k, representing the Nyquist scale, which expresses the minimum separation between the particles, according to the Reference (131).

The overall trend of the power spectra aligns well with the model obtained from CAMB. Particularly, at $a = 0.02$, the spectrum matches exactly with the linear spectrum (and nonlinear

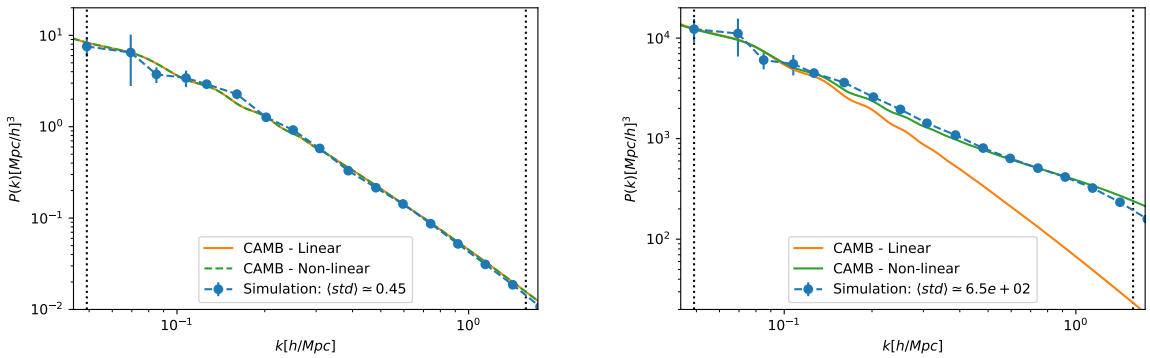

Figure 5 – **DM power spectrum from N-body simulation.** The power spectrum is presented on the left, for $a = 0.02$ (or $z = 49$) and on the right for $a = 1.0$ (or $z = 0$). Both spectra were obtained for a box of side $L_{BOX} = 128$ Mpc$/h$, with $N_p^T = 128^3$ particles, and $N_g^T = 256^3$ cells. The vertical dotted lines indicate the maximum and minimum values for k. They are respectively: $k_{min} \simeq 0.05h/\text{Mpc}$ and $k_{max} \simeq 1.57h/\text{Mpc}$. In both plots we compare the obtained power spectrum to the theoretical linear and nonlinear spectra from CAMB (111).

as well, given the absence of structures at this stage). Conversely, at $a = 1.0$, the resulting spectrum closely resembles the nonlinear spectrum across all scales, indicating the emergence of structure and the breakdown of linear theory on small scales.

2.5.1.8 N-body halo finder, mass function, and bias

We have already seen some details about the halo model in Section 2.3.4. In this model, DM halos are the basic units into which DM particles collapse, starting the process of structure formation. N-body simulations are then used as a data set to what these structures are "identified" with prescriptions that are called *halo finders*. There are many such search methods (including spherical overdensity (SO), friends-of-friends (FOF), and phase-space based algorithms), and we will check some direct effects of some different choices in Section 6.1.1.2 (132, 133). Currently, SubFind (134, 135), Rockstar (136), and VELOCIraptor (137, 138) stand out as the most widely used halo/subhalo finders within the community. A notable distinction among them is that Rockstar and VELOCIraptor incorporates object velocities in its structure identification process.

In this section we will explore the workings of one the most common straightforward halo-finding methods, known as *spherical overdensity* (SO). This method finds spherical objects made up of DM particles that surpass a certain density threshold. This was the chosen approach for halo identification in the work of this section, primarily following the References (98, 99, 101). When applied to a simulation, the SO method finds spherical regions having a density which is expressed as

$$\rho = \frac{N_p^{inside} m_p}{\frac{4}{3}\pi r^3}. \tag{2.93}$$

Here, N_p^{inside} represents the number of particles within the sphere, m_p is the mass of the particles, and r is the radius of the sphere. $\Delta = \rho/\bar{\rho}$ represents the overdensity threshold (here set to 200) used to determine whether or not an object has collapsed. This threshold value is commonly used in the Tinker mass function and halo bias (101, 103).

The general procedure of the SO method can be outlined as follows:

1. Take the positions of the particles and apply the CIC scheme to compute the grid overdensities;
2. Sort the grid cells based on their densities;
3. Select the grid cell with the highest density as the initial candidate center for the first sphere;
4. Begin with an initial sphere radius $r \to r_0$ around this center;
5. Update the center of mass of the particles within this sphere and increase the radius by Δr ($r \to r_0 + \Delta r$) until the density falls below the threshold $\rho = \Delta\bar{\rho}$;
6. Remove all particles found inside the identified halo from the list of particles to ensure no halos are nested within others.
7. Repeat the process by selecting the next grid cell with a higher density.

It is important to note that we set $r_0 = 0.5$ and $\Delta r = 0.1$. Additionally, the process continues until we encounter a grid cell with fewer than 10 particles. Finally, the position, mass, density, and radius of each halo are saved for further analysis.

At redshift $z = 0$ (or scale factor $a = 1$), halos were identified with particle masses of approximately $m_p \simeq 8.6 \cdot 10^{10} M_\odot/h$. The total number of particles found within these halos was approximately $N_{halos} \simeq 1119$, accounting for roughly 20.1% of all particles. Furthermore, the observed halo sizes ranging from an average of $\langle r_{min}^{halo} \rangle \simeq 0.3\,\mathrm{Mpc}/h$ up to a maximum average of $\langle r_{max}^{halo} \rangle \simeq 2.3\,\mathrm{Mpc}/h$.

One of the primary tests conducted on the identified halos involved assessing their halo mass function and bias, as discussed in Section 2.3.4. To validate these measurements, we compare them to a fitting model – in this case we opted to use Tinker's model (101, 103), employing a multiplicity function with a threshold of $\Delta = 200$ (see Equation 2.29). The halo mass function and bias were computed based on the definitions outlined in Equations 2.29 and 2.30, respectively, at $a = 1.0$ (or $z = 0$), considering halos within the mass range $M \in [10^{13}, 10^{14.5}] M_\odot/h$. Overall, the comparison indicates that the identified halos align well with Tinker's fitting functions. However, a slight deviation is notable in the halo mass function for halos at the higher end of the mass spectrum, likely attributable to fewer halos present in those bins.

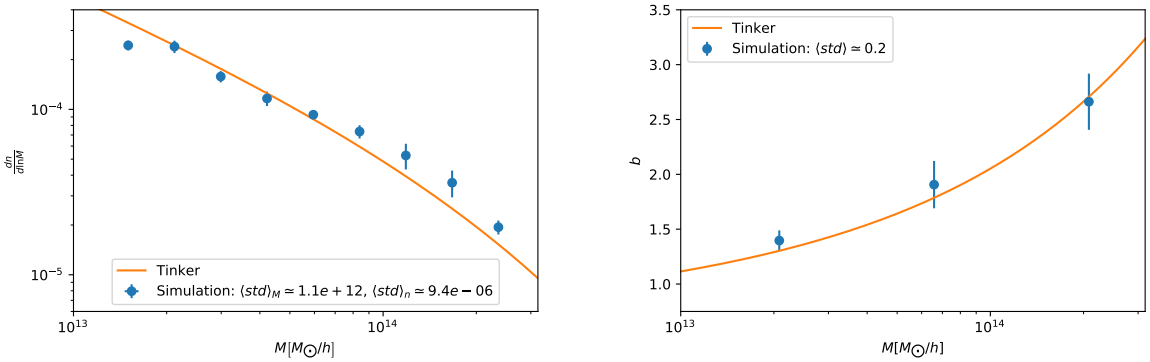

Figure 6 – **Halo mass function and bias from N-body simulation.** On the left: halo mass function. On the right: bias obtained by comparing the spectra of halos and DM. All these quantities were computed at $z = 0$ or $a = 1$ for a box of side $L_{BOX} = 128\text{Mpc}/h$ and mass range within $M_{halo} \in [10^{13}, 10^{14.5}]M_{\odot}/h$, and compared with the Tinker mass function and bias (101, 103).

2.5.1.9 Approximated methods for DM simulations

N-body simulations are the state-of-art of gravitational dynamics for DM particles. However, their computational demands often limit their utility for extensive runs required for comparisons with real surveys, such as parameter estimation and covariance matrices (see Section 2.3.5.1). To address this challenge, numerous approximate methods have emerged to expedite results. Techniques like PTHalos (139), EZMocks (140), PINOCCHIO (141), PATCHY (142), HALOGEN (143), LogNormal (144), ICE-COLA (145), ExSHalos (146), BAM (147, 148), and others aim to generate DM halo catalogs using semi-analytical approximations or by emulating N-body simulations. For a comprehensive review of these methods and their comparison to N-body simulations for power spectrum analysis, we recommend Reference (149).

In Chapter 4 of this thesis, we utilize an approximate method known as ExSHalos (Excursion Set Halos). This approach presents a novel, simple, fast, and parameter-free technique for generating DM halo catalogs (146). Basically, ExSHalos implements the notion of excursion sets (150, 151) and, subsequently, corrects the positions of the peaks using Lagrangian perturbation theory (LPT) (114, 152). The method requires only a fiducial Cosmology, the linear matter power spectrum, and the threshold density for halo formation in linear theory (either constant or ellipsoidal collapse barriers) as inputs.

2.5.2 Hydrodynamical simulations

Hydrodynamical simulations play a pivotal role in comprehending galaxy formation and evolution in the Universe. Unlike DM-only simulations, hydrodynamical simulations incorporate ordinary matter, encompassing all components discussed in Section 2.1. As a result, they serve as the foundation of what we refer to as the *halo-galaxy connection*, directly bridging the information content between two key components: DM halos and galaxy properties. This

integration is essential for deciphering a wide array of observed galaxy characteristics, including spatial clustering, mass distribution, stellar mass, size, color and star formation rate, among other properties (125, 153).

Numerous recent initiatives are pushing the boundaries of hydrodynamical simulations, introducing new variations such as Astrid (154), SIMBA (155), IllustrisTNG (156), Magneticum (157), and SWIFT-EAGLE (158). A remarkable project is the CAMELS (*Cosmology and Astrophysics with MachinE Learning Simulations*) suite, comprising $12,903$ cosmological simulations – $5,164$ N-body and $7,712$ state-of-the-art (magneto-)hydrodynamic simulations. Primarily designed to serve as a data set for ML analyses, CAMELS encompasses all the aforementioned simulations, focusing on small boxes of 25 Mpc$/h$. Additionally, these simulations are utilized in two key contexts in this thesis: halo-galaxy connection (IllustrisTNG) and cosmological parameter inference from galaxy/halo catalogs (Astrid, SIMBA, IllustrisTNG, Magneticum, and SWIFT-EAGLE), detailed in Chapters 5 and 6, respectively.

Indeed, while N-body simulations focus on the gravitational evolution of DM particles, hydrodynamical simulations encompass the evolution of all components, including the gravitational evolution of matter and the hydrodynamical evolution of gas. In some cases, these simulations also account for the interaction of gas with evolving radiation and magnetic fields (153). Initially, the baryon component, representing the visible Universe, consists mainly of gas, primarily hydrogen and helium. Some of this gas material ends up in stars during the process of structure formation. However, at the core of hydrodynamical simulations lie numerical solutions governing ideal, collisional, and non-conducting gases. Modeling the cosmic gas can be approached through three main branches: the Eulerian formulation, the Lagrangian formulation, or a hybrid of both (125). In the Lagrangian formulation, the following equations govern the fluid dynamics

$$\frac{D\rho}{Dt} = -\rho\nabla\cdot\mathbf{v}, \tag{2.94}$$

$$\frac{D\mathbf{v}}{Dt} = -\frac{1}{\rho}\nabla P, \tag{2.95}$$

$$\frac{De}{Dt} = \frac{1}{\rho}\nabla\cdot p\mathbf{v}, \tag{2.96}$$

where $D/Dt \equiv \partial/\partial t + \mathbf{v}\cdot\nabla$ denotes the Lagrangian derivative, ρ is the density, $\mathbf{v}$ denotes de velocity vector, $P = (\gamma - 1)\rho u$ (with γ being the heat capacity ratio and u being the internal energy) denotes the thermodynamic pressure, and $e = u + \mathbf{v}^2/2$ is the total energy per unit mass. This formulation assumes an observer that follows an individual fluid part, specified by its properties such as density ρ, as it moves through space and time. It can also be viewed as a mesh-free technique for approximating the continuum dynamics of fluids by samplings particles (an interpolation of points) (159).

Due to limited numerical resolution of hydrodynamical simulations, which are among the most computationally expensive simulations in Cosmology and Astrophysics, certain

physical processes must be "included by hand". These processes are known as *subgrid physical processes* or sub-resolution models. They bridge the gap between the scales that can be treated numerically, typically above interstellar medium structure scales (around 3 kpc), and those addressed by these subgrid routines, which extend below the scale of star clusters (around 0.3 kpc). It is precisely at this point where the main differences between the different simulations arise, a topic we will explore in Section 6.1.1.1. These subgrid models are a critical component of hydrodynamical simulations, introducing different parameters that must be tuned to ensure that their final products align with observational data (125, 153).

It is not within the scope of the present thesis to delve into all the intricacies of subgrid physical processes. However, we can mention some of them, including:

- **Gas cooling.** This process dissipates the internal energy of gas through mechanisms such as ionization processes. It is often tabulated as a function of density, temperature, redshift, and composition for phenomena like photoionization (160, 161). It is used by EAGLE and IllustrisTNG (153).
- **Element abundance evolution.** This process tracks the time release of individual elements from various nucleosynthetic channels. It is employed in simulations like SIMBA (162), Magneticum (163), EAGLE, and IllustrisTNG (164).
- **Feedback processes.** These processes involve the balance of inflows and outflows that regulate phenomena such as supernovae (SN) and active galactic nuclei (AGN) activity. They may be influenced by mechanisms like stellar winds and radiation pressure.
- Magnetic fields; cosmic rays; dust and others.

These subgrid processes play crucial roles in shaping the evolution of galaxies and the intergalactic medium in hydrodynamical simulations (125, 153).

All these subgrid physical components require calibration, which is typically based on either physical arguments or observations – i.e., calibration aims at reproducing properties of galaxy populations. The most commonly used galaxy property for this calibration is the galaxy stellar mass, which is employed to calibrate feedback associated with stellar evolution. However, in simulations like EAGLE, galaxy size has also been used to reproduce galaxy scaling relations (158). Additionally, properties such as star formation rate and halo gas fractions are utilized in simulations like IllustrisTNG (156). It is important to note that the values chosen for these parameters may vary depending on the resolution of the simulation being considered (153).

In Figure 7 we present a comparison of the galaxy stellar mass function from hydrodynamical simulations, including EAGLE (165), Horizon-AGN (166), IllustrisTNG (167), SIMBA (168), and FIREbox (169), with measurements from the Sloan Digital Sky Survey (SDSS) and

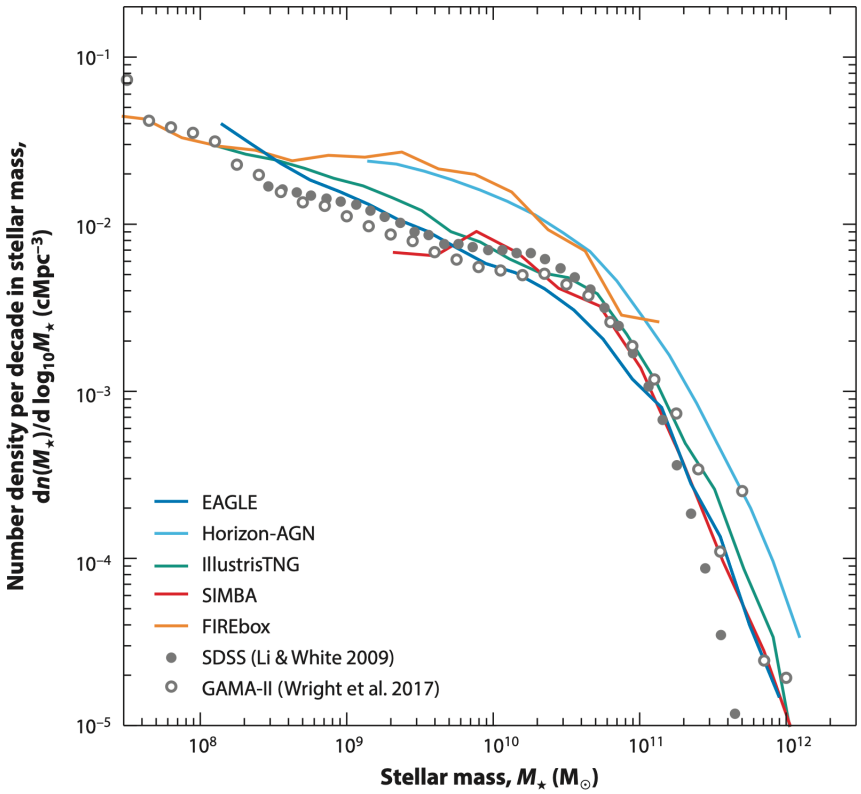

Figure 7 – **Galaxy stellar mass function comparison.** The comparison is done for hydrodynamical simulations such as EAGLE (165), Horizon-AGN (166), IllustrisTNG (167), SIMBA (168), and FIREbox (169) with measurements from SDSS and GAMA surveys (170, 171). *Source:* Reference (153).

Galaxy and Mass Assembly (GAMA) surveys (170, 171). It is important to note that the authors of the figure have omitted the error bars of the real measurements. This plot exemplifies the agreement between simulations, which have box size of $L \simeq 100$ Mpc and mass resolution of $m_g \sim 10^6 M_\odot$, and observational data across the mass range of $[10^8, 10^{11}] M_\star/M_\odot$ for SDSS and GAMA (170, 171). While EAGLE, IllustrisTNG, and SIMBA simulations were calibrated to achieve this level of agreement, success is not guaranteed due to the limited freedom afforded by their subgrid models. Nevertheless, the results are consistent within the highlighted mass range. In contrast, FIREbox and Horizon-AGN simulations exhibit a high number of galaxies at fixed stellar mass. This discrepancy could be attributed to feedback mechanisms (such as AGN) inadequately regulating galaxy growth, poor sampling of energy injection events, or the injection of too little energy per feedback event (153). Consequently, this test serves as compelling evidence of the success of hydrodynamical simulations when compared to real observations.

2.5.2.1 Approximated methods to galaxies and halo-galaxy connection

As we have seen, hydrodynamic simulations represent the best of current simulation capabilities, as they provide a direct reproduction of observable properties of galaxies, which are (for the most part) the objects actually observed in the sky. Unlike DM halos, galaxies

are baryonic entities, making hydrodynamic simulations invaluable for understanding their formation and evolution. However, these simulations are also the most computationally expensive, surpassing even N-body simulations in terms of computational resources required. Given their significance, there has been a concerted effort to develop approximate methods for predicting galaxy properties based on DM halo information. This subfield, often referred to as the *halo-galaxy connection*, aims at extracting galaxy-related information from DM halo simulations (or vice-versa). For a comprehensive overview of these efforts, we recommend consulting Reference (172).

The halo-galaxy connection refers to the relationship between the multivariate distribution of galaxy and halo properties derived from observations and simulations (172). Galaxies form and evolve within DM halos, and many of their properties are intrinsically related to the halo environment and clustering properties. For example, red galaxies tend to populate the centers of halos and are generally older, while blue galaxies are more often found in the outskirts of halos, and are typically younger. Modeling approaches to establish this link generally fall into two categories: physical and empirical models. Physical models include hydrodynamical simulations and Semi-Analytical Models (SAMs), which aim to capture the underlying physical processes governing galaxy formation within halos. On the other hand, empirical models such as Subhalo Abundance Matching (SHAM) and Halo Occupation Distribution (HOD) models are more data-driven and rely on statistical correlations between galaxy and halo properties observed in simulations or surveys.

SAMs approximate various physical processes using analytic prescriptions that can be tracked through the merger history of DM halos, and many codes are examples of this idea, such as Santa Cruz (173), GAEA (174), and L-Galaxies (175). SHAMs, on the other hand, establish a relationship between the mass of a galaxy and the abundance of the DM halos it typically inhabits (176). Finally, Decorated Halo Occupation Distribution models (decorated HODs) introduce additional halo properties besides mass, such as concentration, to determine the probability density distribution for the number of galaxies within their hosting halos (177).

In the present thesis we make use of the IllustrisTNG hydrodynamical simulation to give our contribution for the halo-galaxy connection. We achieve this by presenting a new category of solutions, using ML techniques in order to reproduce galaxy properties based on halo information. This work will be presented in Chapter 5.

3 MACHINE LEARNING BACKGROUND

This thesis is also firmly grounded upon the principles of Machine Learning (ML), which serves as the primary set of techniques utilized throughout the work presented here. This chapter provides a concise overview of various ML methods, laying the basis for the developments of subsequent chapters. For a more comprehensive understanding of the topics covered here, we recommend References (41, 178, 179).

We begin with a brief review of fundamental ML concepts in Section 3.1. Section 3.2 presents some traditional ML approaches. k-Nearest Neighbors (kNNs) are seen in Subsection 3.2.1, as an example of clustering methods. Tree methods are presented in Subsection 3.2.2, including Extreme Randomized Trees (ERTs) and Light Gradient Boosting Machines (LGBMs). Symbolic Regression is addressed in Subsection 3.2.3. Transitioning to Deep Learning (DL), Section 3.3 introduces key concepts for all neural networks (NNs) seen through the thesis. As the first examples of NNs, we present the seminal idea of Multi Layer Perceptrons (MLPs) in Subsection 3.3.1. Expanding on the foundation laid by MLPs, Section 3.3.2 introduces Convolutional Neural Networks (CNNs) and their constituent blocks. In Section 3.3.3, we explore image denoising techniques utilizing CNNs. Section 3.3.4 focuses on Graph Neural Networks (GNNs), also rooted on MLPs, covering topics such as graph definition and construction, GNNs layers, and what we coin as "GNNs variations", including also a non GNN architecture, the deep sets. Alternatives to the usual maximum-likelihood algorithms are found in Section 3.4, with the probabilistic methods, where we discuss Moment Neural Networks (MNNs) and Regression to Classification, presenting NNClass algorithm. Finally, Section 3.5 addresses the challenge of imbalanced data sets, techniques for combining different ML predictions, and concludes with a discussion on hyperparameter search strategies.

3.1 From the beginning: machine learning notions

In essence, classical programming derives answers based on predefined rules and data. Artificial intelligence (AI) encompasses the field of computer science dedicated to creating systems that can perform tasks that typically require human intelligence. Within AI, ML is a subset focused on developing algorithms and models that enable computers to learn patterns and make predictions or decisions based on data without being explicitly programmed. In other words, ML formulates rules based on data and can generalize these rules to handle unseen samples. The term "machine learning" is often attributed to Arthur Samuel, with his research about the game of checkers (180). Central to the motivation behind ML is the question: "How can computers learn to solve problems without being explicitly programmed?". Consequently, ML emerges as a tool specifically designed to tackle large and intricate data sets, uncovering nonlinear relationships within high-dimensional feature spaces (41).

Traditional ML involves algorithms trained on *structured data*[1], often requiring manual feature engineering to extract relevant patterns. Deep learning (DL), a subfield of ML, employs artificial neural networks, or simply neural networks (NNs), with multiple layers (hence "deep") to automatically learn hierarchical representations of data directly from raw input, eliminating the need for manual feature engineering. While DL offers advanced capabilities for learning complex patterns, traditional ML algorithms remain vital for structured data and tasks where interpretability or small data sets are essential.

ML tasks can be broadly categorized into three main types: *supervised*, *unsupervised*, and *reinforcement learning* (41,179). In the first two cases, these methods aim to learn a function $\mathbf{F}(\mathbf{x} \rightarrow \mathbf{y})$ from input data $\mathbf{x}$ to output data $\mathbf{y}$. Supervised learning involves learning the relationship between input data $\mathbf{x}$ and output data $\mathbf{y}$, where examples of output data are provided for learning. On the other hand, unsupervised learning aims at uncovering patterns and features present in the input data $\mathbf{x}$ without access to corresponding output data $\mathbf{y}$. While reinforcement learning involves agents learning optimal behavior by interacting with an environment, receiving feedback in the form of rewards or penalties for their actions. Each method addresses different learning scenarios: supervised learning for labeled data prediction, unsupervised learning for pattern discovery in unlabeled data, and reinforcement learning for decision-making in dynamic environments.

Within the realm of supervised learning, tasks are further subdivided into *regression* and *classification*. Regression tasks involve predicting a continuous variable for the output data $\mathbf{y}$, while classification tasks entail predicting a discrete variable. In all cases, these methods are designed to handle (N, M)-dimensional data for both input $\mathbf{x}$ and output $\mathbf{y}$, respectively. It is quite clear, then, that the larger these dimensions are, the harder it is to train the model.

A common practice in supervised ML tasks is to partition the input $\mathbf{x}$ and output $\mathbf{y}$ data sets into *training*, *validation*, and *testing* sets. These sets are utilized in the corresponding stages of the learning process: *training*, *validation*, and *testing*. During the training stage, the algorithm learns the underlying patterns and rules using only a portion of the data set known as the training set. Simultaneously, to monitor the training progress and ensure the model can generalize well to unseen data, a separate fraction of the data set, the validation set, is used. This validation set serves to evaluate the performance of the model on data it has not been trained on. Finally, the performance of the trained algorithm is assessed on yet another independent portion of the data set, known as the testing set, used to benchmark the model. Various metrics and scores can be calculated at this final stage, depending on the nature of the problem at hand, such as mean squared error for regression tasks, and accuracy for classification tasks. In addition to organizing the data set into training, validation, and testing sets, each of these sets can be further subdivided into smaller portions called *batches*. Batches represent subsets of the

[1] Structured data refers to data that is organized in a well-defined manner, typically arranged in rows and columns, e.g., tabular data.

data used to train the model incrementally, with the model's internal parameters updated after processing each batch of samples. This *batch-wise training approach* helps improve efficiency and scalability, particularly for large data sets.

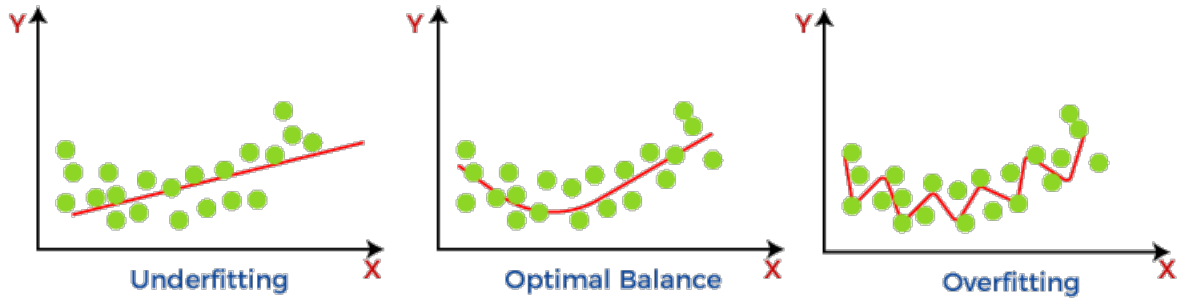

Figure 8 – **Examples of underfitting (left panel), balanced model (middle panel), and overfitting (right panel).** The idea is to assess the ability of the model (red curve) to describe the data set (green dots). In the case of underfitting, the model oversimplifies the data set, providing a poor fit that fails to capture the underlying pattern exhibited by the data points. Conversely, in the optimal balance scenario, the model aligns well the data, capturing its general trend without overcomplicating the representation. However, in the case of overfitting, the model conforms excessively to the data, attempting to pass through each data point. *Source:* Analytics Vidhya (181).

The primary objective of monitoring the algorithm performance on both training and validation sets (and, subsequently, on the testing set) is to identify "ill models". Specifically, we aim to assess the algorithm's ability to generalize its predictions beyond the training set by evaluating its performance on the validation and testing sets and comparing the results. *Overfitting* occurs when a model learns to capture noise or random fluctuations in the training data, rather than the underlying patterns or relationships. As a result, the model performs well on the training data but fails to generalize to the validation or testing sets. Conversely, *underfitting* occurs when a model is too simplistic to capture the underlying structure of the data, resulting in poor performance on the training, validation, and testing sets. A visual representation of these phenomena, depicting underfitting, optimal balance, and overfitting, can be observed in Figure 8. While there are various approaches to address these issues, the general strategy involves adjusting the model's complexity based on the amount of training data available. In summary, by carefully monitoring algorithm performance and identifying cases of overfitting or underfitting, we can refine our models to achieve better generalization and more accurate predictions on unseen data.

Another important aspect of ML algorithms, which will be emphasized in Chapters 4 and 6, is *robustness*. While it is a common practice to evaluate the generalization power of a model, ensuring it can effectively handle unseen samples from the same data set used for training (on the validation and testing sets), it is equally important to measure their ability to extrapolate predictions to different data sets that share similar properties with the training set. These data sets may vary due to factors such as the underlying physical model used to generate

them or some evolution of samples over time. Unfortunately, ML models often struggle with this extrapolation task due to inherent differences across data sets. To address this issue, one common approach is to retrain the model on a larger data set that includes the new samples. In Cosmology, where different simulations or unique realizations of the Universe are prevalent, it is crucial to develop models that exhibit robustness across a wide range of data sets. This robustness enables the model to generalize effectively to various simulation scenarios and, eventually, extrapolate predictions to real observational data. By aiming for robustness, we can enhance the reliability and applicability of ML techniques in cosmological studies, facilitating insights into the underlying physics of the Universe.

However, in all relevant applications of ML techniques, even the most robust and high-performing methods are not perfect. In ML, errors in predictions can be categorized into two main types: *epistemic* (systematic) and *aleatoric* (statistical) uncertainties. The epistemic error stems from the inherent limitations of the model itself, which propagate to its predictions. Since it arises from an incomplete knowledge or understanding of the complexity of the data set as captured by the model, it can often be reduced by increasing the size of the training data set. In other words, epistemic errors reflect uncertainties associated with the model's parameters and structure, and as more data is accrued, the model's predictive capability improves, and those uncertainties tends to decrease.

Unlike epistemic uncertainties, aleatoric uncertainties arise from inherent randomness (or variability) in the data itself. It is associated with measurement errors, or some inherent stochasticity in the observed phenomena. Aleatoric uncertainties cannot be eliminated by increasing the size of the data set: they persist regardless of the amount of data available for training. Therefore, aleatoric errors reflect the intrinsic statistical fluctuations or variability in the observed phenomena.

By considering both epistemic and aleatoric uncertainties (resulting in the total predictive uncertainty for each sample), ML practitioners can gain a more comprehensive understanding of the predictive capabilities and limitations of their models.

Throughout this thesis, we focus on supervised learning algorithms tailored for regression tasks. Our approach encompasses various methodologies across different chapters. In Chapter 4 we employ a computer vision technique – a denoising autoencoder – as an image denoiser to tackle regression tasks. This involves utilizing the autoencoder architecture to reconstruct clean images from noisy inputs. In Section 5.3 we explore multiple regression methods and employ ensemble techniques, such as stacked ML models, to combine them. Additionally, we compare their performance on both the original data set and an augmented data set. This augmentation forces the models to make predictions in under-represented regions of the data set. In Chapter 6 we delve into the use of graph neural networks (GNNs) for performing global inferences. These GNNs are specialized neural network architectures designed to operate on graph-structured data, allowing us to extract insights from complex relational data sets. In

each of these methodologies, we make use of the fundamental principles of ML outlined in this section. By leveraging these principles, we aim to develop robust models capable of effectively addressing regression tasks across various domains and data sets.

3.2 Traditional Machine Learning Methods

In this section we make an overview of some of the traditional ML approaches. We commence with one of the simplest algorithms, the k-Nearest Neighbors (kNN), in Section 3.2.1. Following that, we introduce the most easily interpretable methods, the tree methods, in Section 3.2.2. Here, we elaborate on Random Forests, Extreme Randomized Trees (ERTs), and Light Gradient Boosting Machines (LGBMs). Then, in Section 3.2.3, we delve into Symbolic Regression (SR), a valuable method for interpretability.

3.2.1 k-Nearest Neighbors

The k-Nearest Neighbors (kNN) algorithm is a non-parametric learning method that calculates the distance from a new data point to all other training points, assigning the point to the class to which the majority of the k neighbors belong. Specifically, given a matrix X with dimensions $(N \times D)$, being N the number of points and D is the number of dimensions, and x_i represents the i-th point of X with D components, the *Euclidean distance* is computed as follows

$$\Delta(x, x_i) = \sqrt{\sum_{d=1}^{D} (x_d - x_{i,d})^2}. \tag{3.1}$$

The purpose is to find the value $x^\star = \operatorname{argmin} \Delta(x, x_i)$.

kNN is predominantly used as an unsupervised ML method, i.e., as a *clustering algorithm*, but can also be framed as a supervised ML method to tackle classification and regression problems. In the case of classification, it performs "voting" step to determine the class label of the new instances. While in regression problems, the final predictions is given by local interpolation (or averaging) of the targets associated with the nearest neighbors (41, 182). This method has been used in References (183, 184) for studying the halo-galaxy connection. In this thesis, we employ the SkLearn K Neighbors Regressor library (182).

3.2.2 Tree methods

Tree methods are commonly employed for classification tasks, but they also have regression variants, both based on *Decision Trees* (DTs) (41, 182). The name stems from the model's representation, which adopts a "tree structure" (see Figure 9). The initial *node*, known as the *root node*, contains the entire data set. At each *branch* of the tree, the data is partitioned into two *child nodes*, or subsets, based on a predefined *decision boundary*. One node holds data below the boundary, while the other contains data above it. This splitting process recurs until a predefined stopping criterion is met. The *leaf nodes* represent the final decision or prediction

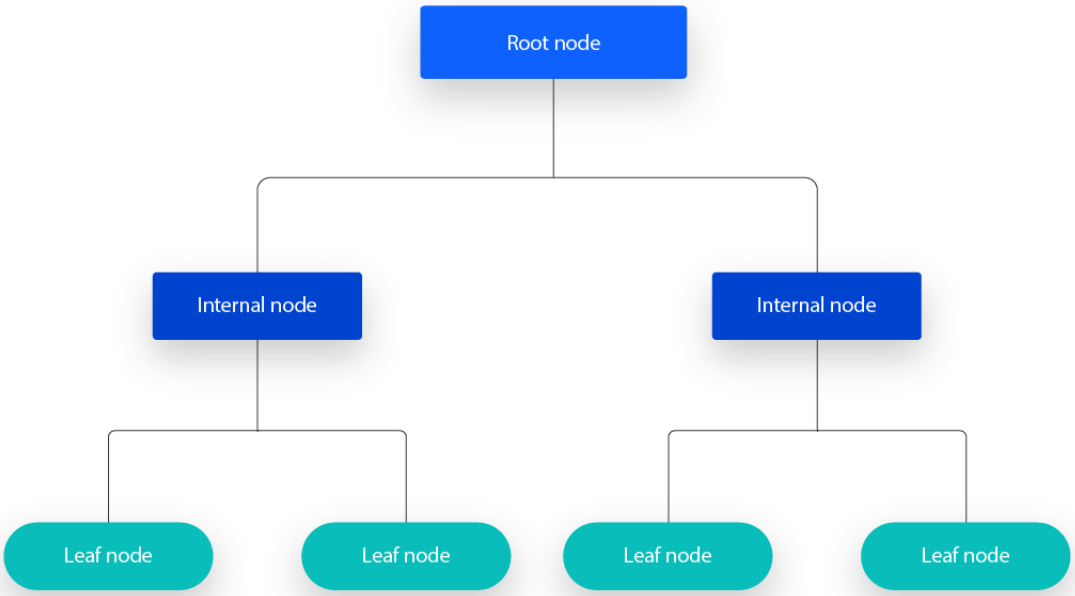

Figure 9 – **Scheme of a decision tree.** It starts from the root node, achieves the internal nodes, and ends up in the leaf nodes. *Source:* IBM (185).

of the model. In regression problems, the predictions are averaged over discrete responses, approximating a continuous function.

DTs are a straightforward method that is intuitive to visualize and interpret. They naturally reflect the way we might analyze a data set manually, employing a hierarchy of increasingly detailed questions. Additionally, DTs serve as the foundation for other tree-based methods, which we will briefly discuss in this section.

A DT is constructed by selecting a feature and defining a value to split the data. If we base this criterion on the information content of the data set, we can define the *entropy* E as

$$E(x) = -\sum_i p_i(x) \ln \left[p_i\left(x\right)\right], \tag{3.2}$$

where x represents the data set, i is the class, and $p_i(x)$ is the probability of that class given the training data. Another useful quantity defined in terms of this criterion is the *information gain* IG, which corresponds to the reduction in entropy due to the split. Essentially, it represents the difference between the entropy of the parent node and the sum of entropies of the child nodes

$$IG(x) = E(x) - \sum_{i=0}^{1} \frac{N_i}{N} E(x_i), \tag{3.3}$$

where we consider a binary split with $i = 0$ representing the points below the split criteria, and $i = 1$ representing the ones above it. N_i is the number of points x_i in the i-th class, and $E(x_i)$ is the entropy for that class. Then, the value of the feature to split the data (the threshold) is defined by maximizing the information gain for a given split point s. Other splitting criteria, such as the *Gini coefficient* or *Gini impurity*, can be used in a similar fashion.

If a stopping criterion is not defined, the tree could continue growing until there is only a single point per node in the splitting set. This is highly computationally expensive and usually leads to overfitting. Therefore, the stopping criterion is often defined as pruning the tree or limiting its depth, thus also limiting its complexity.

There are two main derivations of DTs that employ *ensemble learning*, which consists of combining the outputs of multiple models:

- **Bagging.** This method averages the predictive results of a series of *bootstrap samples* (a randomly drawn subset of a data set created by sampling with replacement) from a training set of data. Specifically, for a training set x of N samples, bagging generates K equally bootstrap samples, with some estimate function $f_i(x)$. The final estimator defined by bagging is
$$f(x) = \frac{1}{K}\sum_{i}^{K} f_i(x). \tag{3.4}$$
This ensemble approach often leads to more robust and accurate predictions compared to a single model.

- **Random Forests (RF).** They expand the concept of the bootstrap and bagging by generating a set of DTs from the bootstrap samples (186). The features to generate the tree are selected randomly from the complete set of features in the data. Additionally, they consider a random subset of features at each split instead of considering all features and selecting the best one per split, as in usual DTs. The final task, classification or regression, involves determining the majority or averaging the results of the individual DTs, respectively. This method addresses two limitations of DTs: (i) overfitting the data, as trees are now shorter, and (ii) limited extrapolation capability, as it explores the correlation of different features and nonlinear decision boundaries by combining the predictions of multiple decision trees, each trained on different subsets of data and features in data sets.

Nowadays, we have many other variations of DT methods, which allow improvements in model expressiveness and result in faster models. This is related to the fact that many applications of DTs date back to the early 2000s. For instance, Reference (187) employs DTs with bagging on SDSS data. Meanwhile, we have many applications of RF, as sees in the References (64, 188), in the context of halo-galaxy connection. There are many other methods built upon the basic ideas of DTs, and we will present some of them in the next sections.

3.2.2.1 Extreme Randomized Trees

Extreme Randomized Trees (ERTs) comprise an ensemble method where individual "weak" learners (in this case, DTs) are combined to build a powerful estimator. They sample the entire data set and randomize the splitting process of the individual DTs, making them

different from RF and faster (due to this random nature). While a single DT is likely to overfit, their random nature reduces the variance of the estimator. The final prediction of the model is the average over the predictions from all individual DTs (189).

ERTs have already been successfully employed in the context of halo-galaxy connection studies (53, 56, 190). In this thesis, we use the SkLearn ensemble Extra Trees Regressor library (182).

3.2.2.2 Light Gradient Boosting Machines

Light Gradient Boosting Machines (LGBMs) represent a gradient boosting framework that implements gradient boosting decision trees (GBDTs) (191). Gradient boosting means implementing a "boosting" technique where "weak learners" (simple DTs) are transformed into "strong learners" by minimizing a *loss function*[2] associated with the weights given by the leaf nodes of multiple DTs (192). In this scenario, the trees do not grow independently; each new tree is designed to improve the previous one based on knowledge learned from the previous trees (previous leaf scores). The reason for the word "light" in the name comes from the *leaf-wise* method used to grow the trees. This means the algorithm chooses the leaf with the maximum variation in the loss to grow. Additionally, it makes use of other strategies to reduce the number of features and samples.

LGBMs have been used in a variety of applications, including in Astronomy and Astrophysics (193–196). Here we have used the LightGBM package.

3.2.3 Symbolic Regression

While NNs excel at providing precise and accurate approximations of complex data relationships, interpreting them can be challenging due their use of a large number of parameters, rendering them as "black boxes" in many cases. Therefore, there is a need to extract mathematical expressions that describe or approximate the relationships learned by NNs, as understanding these relationships in such forms can be more straightforward. *Symbolic regression* (SR) is a ML algorithm specifically designed to discover *symbolic expressions* that fit data from a function. This can be achieved in two ways: by directly deriving a symbolic expression from the correlation between the input data and output data of a data set, or by training a ML algorithm to uncover this latent representation (which can involve a variety of methods, including MLPs, GNNs, or others) and subsequently translating it to an equation (197, 198).

The primary objective of these algorithms is to utilize an optimization framework to minimize both prediction error and model complexity. Specifically, SR methods employ *genetic*

[2] A loss function is a mathematical measure to quantify the discrepancy between the predicted values of a model and the actual ground truth values in a data set. It represents the penalty incurred by the model for making incorrect predictions. We will see a better definition of loss function, with examples, in Section 3.3.1.

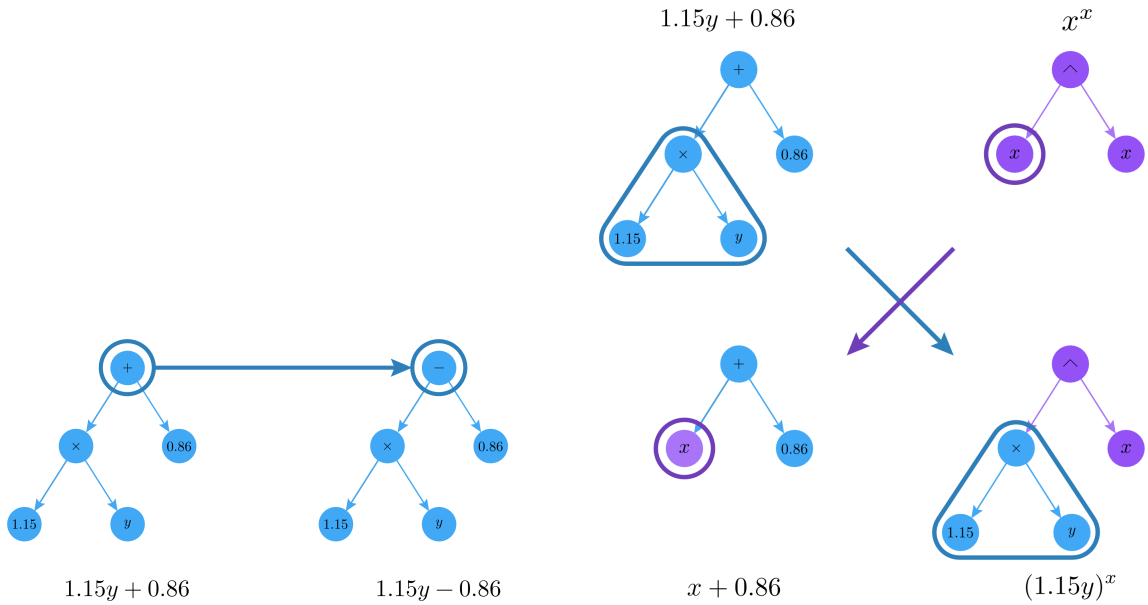

Figure 10 – **Scheme of *mutation* (left panel) and *crossover* (right panel) operations into a symbolic regression algorithm.** Mutation involves replacing one operator, such as $+ \rightarrow -$ in a tree structure, while crossovers mix branches of two trees, bringing the operators and values together to the other tree. *Source:* Reference (198).

programming, which searches for the optimal analytical expression by creating combinations of given *operators* and input variables arranged in a *tree* structure. During training, the regressor can utilize standard mathematical operators such as: "ADD", "SUB", "MULT", "DIV", "POW", "ABS", "LOG", "LOG10", "SQRT", "1/X" (representing addition, subtraction, multiplication, division, exponentiation, absolute value, natural logarithm, base-10 logarithm, square root, and inverse of a variable, respectively). Subsequently, a standard loss function is selected to optimize the fitting process.

The expressions discovered in each generation are evaluated, and the most accurate ones are preserved to the next generation. This iterative process involves *mutations* and *crossovers* to explore the entire equation space and find an accurate expression. Mutation involves replacing an operator in a tree structure, while crossovers entail mixing branches of two trees, replacing one part of the operation with another operator (see Figure 10 for a comprehensive example).

During training, the algorithm produces a list of equations identified by the regressor. For each equation, usual SR algorithms provides three metrics to assess its fit: *complexity*, an user predefined *metric*, and a *score*. The *complexity* of the equations accounts for the number of operators, constants, and variables utilized. The *metric* and the complexity are combined to generate an overall score for the equations. The algorithm arranges the equations from the least to the most complex. Then, for each equation, it computes the fractional decrease in complexity, relative to the next one. The *score* is maximized if this fractional decrease is large. The *symbolic expression* is selected by evaluating multiple candidate equations on a test set, aiming to optimize the trade-off between complexity and accuracy.

Through this work we have used PySR library (199) for SR applications.

3.3 Deep Learning Methods

This section is devoted to dive into some of the deep learning (DL) algorithms used in the present thesis. First, we describe in details the simplest neural network, the Multi Layer Perceptron (MLP), in Section 3.3.1. Second, in Section 3.3.2, we explain about convolutional neural networks (CNNs), accounting for image denoising techniques in Subsection 3.3.3. Third, we present the networks specialized to deal with graph data, the graph neural networks (GNNs), in Section 3.3.4.

3.3.1 Multi Layer Perceptrons

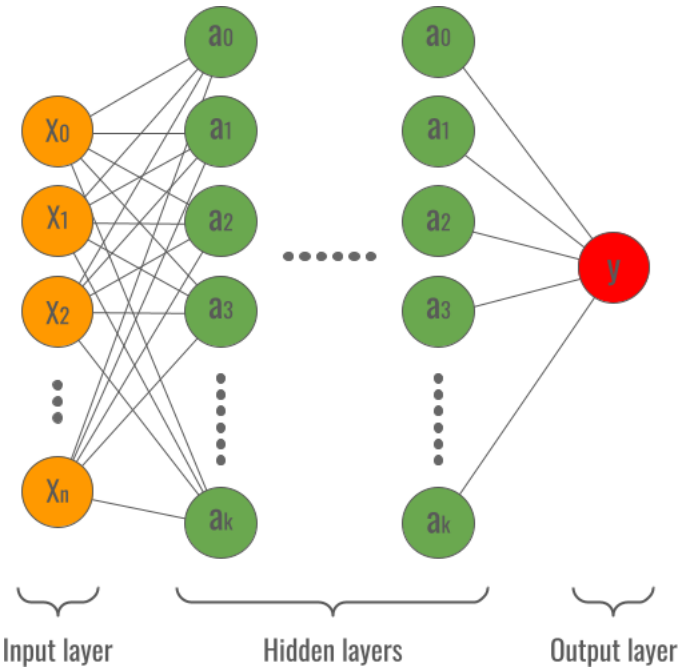

Figure 11 – **Representation of a MLP.** Each circle represents a neuron, lines symbolize the connections between the neurons, and each vertical sequence denotes a layer. Each connection is associated with a weight represented by ω_i. Each neuron carries its activation value denoted as x_i (input value), a_i (hidden values), and y_i (output value). The first layer (represented in the figure by the orange neurons) carries on the input vector (with $\mathbf{x}$). This is followed by the hidden layers (shown in the figure as the columns of green neurons), which perform the general transformation $f_\mu\left(b_\mu + \sum_\nu \omega_{\mu\nu} a_\nu\right)$ (see Equation 3.5). At the end, the output layer (represented by the red neuron) delivers the output vector $\mathbf{y}$.

Multi Layer Perceptrons (MLPs) draw inspiration from biological neural systems, such as the brain, to process information. These algorithms propagate information through a layered architecture composed of activation units called "neurons" and their connections (see Figure 11). Each neuron maintains its activation state, while the connections carry weights that multiply the input signals as they propagate through the network. The first layer, known as the *input layer*,

receives the initial input values (denoted as $\mathbf{x}$), and subsequent layers process this information through transformations. The final layer, known as the *output layer*, produces the network's prediction (denoted as $\hat{\mathbf{y}}$). The training process of the neural network involves learning the optimal values for the connection weights (denoted as ω_i). These weights collectively form a weighting matrix (denoted as W), which encapsulates the learned information and is utilized to make predictions on new data samples (41, 178, 179). In MLPs, layers are often referred to as *dense layers* (also known as *fully connected layers*), and the MLP architecture is typically structured as a sequence of such layers (200–202).

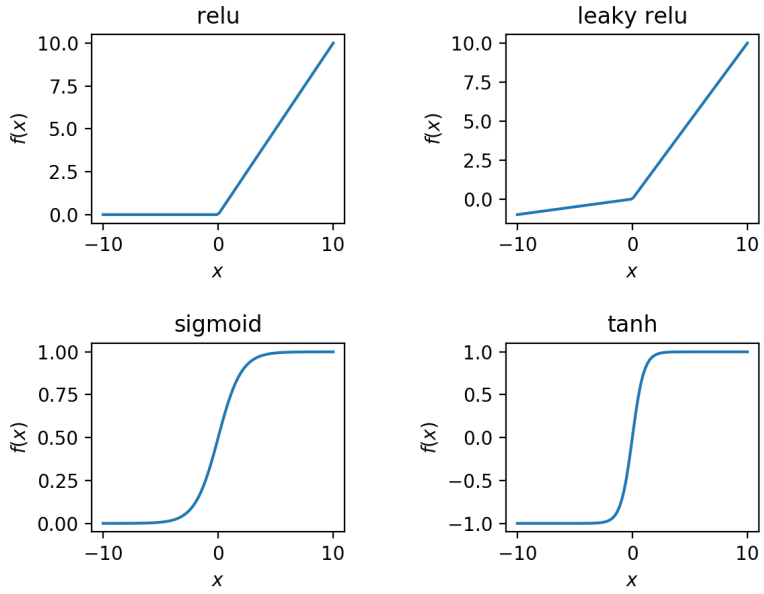

Figure 12 – **Representation of different activation functions.** Rectified Linear Unit (ReLU), Leaky Rectified Linear Unit (Leaky ReLU), Sigmoid (Sigmoid) and hyperbolic tangent (Tanh).

Basically, each neuron in the hidden layers within a MLP carries the neuron value a_i and each connection is characterized by a weight ω_i. Then, each hidden layer is responsible by performing a transformation in the neurons values. This transformation is composed by two parts: the application of an *activation function* $f_\mu(\cdot)$ and the addiction of a bias b. The activation function is a nonlinear transformation that can assume different forms. Common activation functions include Rectified Linear Unit (ReLU), Leaky ReLU, Sigmoid, and hyperbolic tangent (Tanh) – see Figure 12. These functions determine whether the neuron's information is transmitted to the next layer based on the functional form of the function, thereby activating or deactivating the neuron. Additionally, every hidden layer μ includes a bias vector b_μ, which is a linear and an additive parameter, which is added to each neuron in the layer. It serves to shift the output values of each neuron, introducing a linear degree of freedom to the layer transformation. Then, we can define a input array for each layer as $\mathbf{a}_\mu = (a_{\mu,0}, a_{\mu,1} \ldots, a_{\mu,k}) \in \mathbb{R}^{k+1}$ and a weight matrix $W \in M_{m\times(k+1)}(\mathbb{R})$, given an intermediate output array $\mathbf{y}_\mu \in \mathbb{R}^m$ for each layer

μ as

$$y_\mu = f_\mu \left(b_\mu + \sum_\nu \omega_{\mu\nu} a_\nu \right). \tag{3.5}$$

The primary objective of these transformations is to enhance the nonlinearity of MLP predictions (41, 178, 179). We can have as many layers m as we wish like, but the price may be an increasing complexity of the network. The same argument applies for the number of neurons k per layer (which, in principle, can be different for each hidden layer) (200–202).

Before moving on to the optimization process, it is important to address the initialization of the network weights. At the very start of the training, the weights of the network are set using a process known as *initialization*. This step involves defining the initial weights either with constant values or by sampling from a specified distribution (200–202). These initial weights provide the starting point for the optimization process, which seeks to adjust the weights in a manner that minimizes the network's loss function and improves its performance.

The optimizer plays a crucial role in finding the weights such that a certain function, called the *loss function* $\mathcal{L}$, is minimized. The loss function quantifies the disparity between the predicted values $\hat{\mathbf{y}}$ and the true values $\mathbf{y}$. In regression tasks, a commonly used loss function is the *mean squared error* (MSE), given by

$$\text{MSE} = \frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2. \tag{3.6}$$

This function is designed to always yield non-negative values, with smaller values (closer to zero) indicating better performance. By computing the squared differences, the MSE penalizes large disparities between y_i and $\hat{y}_i$ more severely than smaller differences. Hence, the model is incentivized to minimize larger errors while still addressing smaller discrepancies. Alternative loss functions, such as the *Mean Absolute Error* (MAE) for regression, or *categorical cross-entropy* for multilabel classification problems, can also be defined depending on the requirements of the specific task at hand (200–202).

Different optimizers, such as Adam (Adaptive Moment Estimation), SDG (Stochastic Gradient Descent), RMSProp (Root Mean Square Propagation), and others, share the fundamental principle of searching through the parameter space of weights until reaching a (hopefully global) minimum for the loss function (41, 178, 179). This process involves taking iterative steps, often referred to as *training epochs*, where the loss function is computed and evaluated. The goal is to determine whether the loss decreases with each step or not. Training continues for a defined number of epochs, or until a stopping criterion is met – e.g., when the error falls below a certain threshold. The size of each step is determined by a parameter known as the *learning rate* (denoted as ℓ).

In the case of the foundation optimization algorithm, known as *gradient descent*, the weights are updated according to the derivatives of the loss function with respect to the network

parameters:

$$W_{ij} \rightarrow W_{ij} - \ell \frac{\partial}{\partial W_{ij}} \mathcal{L}(W|a, \hat{y}). \tag{3.7}$$

An *optimizer* refers to a specific variant or extension of the gradient descent algorithm that incorporates additional features or modifications to improve its performance or efficiency (as the ones mentioned previously). In the networks developed for this work, the Adam optimizer was primarily used. This optimizer incorporates a technique called momentum, which utilizes fractions of the updated parameter vector to adjust the current one. For further details on the Adam optimizer and other optimization algorithms, we refer the reader to References (200–204).

To find the best configuration of weights, MLPs utilize a technique called *backpropagation* (205). This method involves the following steps:

1. *Perform forward propagation:* process the input data through the network to generate predictions.
2. Compute the difference between the true labels and the predictions for the last layer.
3. Backpropagate this difference through each hidden layer, computing the error gradients for each layer.
4. Evaluate the partial derivatives of the individual errors per layer with respect to the weights per layer.
5. Combine the errors to obtain a total gradient related to the weights of the network.
6. Update the weights according to the learning rate and the total gradient, similar to the gradient descent algorithm.

Backpropagation allows the network to adjust its weights based on the error calculated during forward propagation, gradually improving its performance over multiple iterations. For a detailed explanation of this method, see References (178, 179).

MLPs are extensively employed for solving regression and classification tasks involving *tabular data*, where there exists a correlation between the input data $\mathbf{x}$ and the output data $\mathbf{y}$. For example, in Reference (42), MLPs are utilized for making inferences on cosmological and astrophysical parameters using various summary statistics as input data, while in Reference (61), they are employed to predict total subhalo mass based on other properties of the subhalos. Additionally, MLPs were utilized in the present thesis in Section 5.3 (where we are specifically naming them as neural networks (NNs)) and Section 5.4, and throughout Chapter 6. It is important to note that the libraries used in the thesis to implement the MLPs were Keras (200) and PyTorch (202), which are widely adopted frameworks for building and training MLPs in Python.

3.3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are networks specifically designed to handle matrix data, more specifically regular data[3]. Images are a prime example of such data, represented as matrices of pixels, with dimensions for height, width, and color channel (typically, RGB images have three color channels corresponding to red, blue, and green). This feature has made CNNs immensely popular in computer vision applications (179). In this section we will explore the fundamental building blocks of CNNs, and in the subsequent section we will delve into how they can be effectively utilized in image denoising algorithms.

3.3.2.1 Convolutional Neural Network Blocks

The term CNN is applied to architectures that include at least one *convolutional layer*, although they often incorporate various other components (different kinds of layers), each one with a different purpose (200–202).

3.3.2.1.1 Convolutional Layers

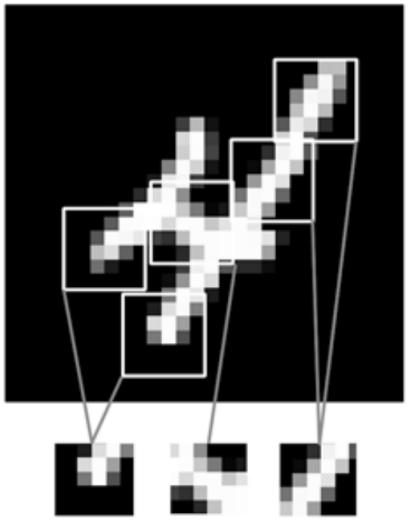

Figure 13 – **Visualization of the local patterns learned by convolutional layers.** We can see the handwritten digit 4 with parts of it extracted by small square windows. *Source:* References (179, 206).

The operation enacted by convolutional layers is akin to that of dense layers, but with a crucial distinction. Whereas dense layers are specialized to discern *global patterns* within their feature space (e.g., patterns involving all vector entries or all image pixels, if compressed into a 1D array), convolutional layers specialized in detecting *local patterns*[4]. These patterns

[3] Images are regarded as regular data due to their structured representation as two-dimensional grids of pixels with fixed positions and homogeneous channels.

[4] Note that this does not mean that dense layers can not learn local patterns. Depending on the architecture and the amount of data available to train the model, dense layers can be used to learn them.

are identified within small 2D windows of an image, known formally as *kernels*. For example, in Figure 13 we observe a handwritten[5] image of the digit 4 with certain segments highlighted by small 2D windows (the kernels). The fundamental concept is that these convolutional layers can recognize the image as representing the digit 4 based on its edges, textures, and other localized patterns. Essentially, by segmenting the image into small sections, the kernels are tasked with capturing the entirety of its visual information (179).

Convolutional layers are adept at learning patterns within specific regions of images, thereby incorporating *spatial translation invariance*. This means that once a pattern is learned, networks featuring these layers can recognize it irrespective of its position within another image. Moreover, convolutional layers facilitate the creation of a hierarchy among these learned patterns. As a result, different layers become specialized in detecting distinct patterns, ranging from various parts (of different sizes) of an image to different textures and features (179).

To be precise, 2D convolutional layers are characterized by two essential parameters:

- **Number of filters.** These are the units responsible for learning various characteristics from the images (edges, textures, or shapes). The number of filters determines the depth of the output volume produced by the convolutional layer. Each filter learns to detect different features or patterns in the input data.

- **Kernel size.** This refers to the dimensions of the small windows or patches responsible for extracting patterns from localized regions of the input image. In other words, the kernel size specifies the spatial dimensions of the filters.

These convolutional layers transform an input image with dimensions $(N \times M \times C)$, where C represents the color channels, into structures with a new set of dimensions determined by the *convolutional operation*. This operation involves a *dot product* between the kernel and the input data. The movement of the kernel across the input data is determined by a parameter known as the *stride*, which specifies how many pixel units the kernel shifts at each step. This process generates the final output matrix or *feature map*. For instance, if we have an input matrix with dimensions $(6 \times 6 \times 3)$, a (3×3) kernel, a stride of (1×1), and 64 filters, the resulting feature map would be $(4 \times 4 \times 64)$. Adjusting the stride parameter allows for the effective reduction of the dimensions of the 2D output data.

In Figure 14 we show an input matrix represented as a feature map with dimensions $(5 \times 5 \times 2)$, where (5×5) denotes the spatial dimensions and 2 represents the depth (or number of channels). The convolutional layer applies a set of kernels to this input matrix. These kernels are smaller patches with dimensions (3×3) in this example. As the kernels move across the

[5] The image shown in Figure 13 was taken from a library of handwritten digits, the MNIST (Modified National Institute of Standards and Technology) data set (206), which is one of the most popular databases for image applications in ML.

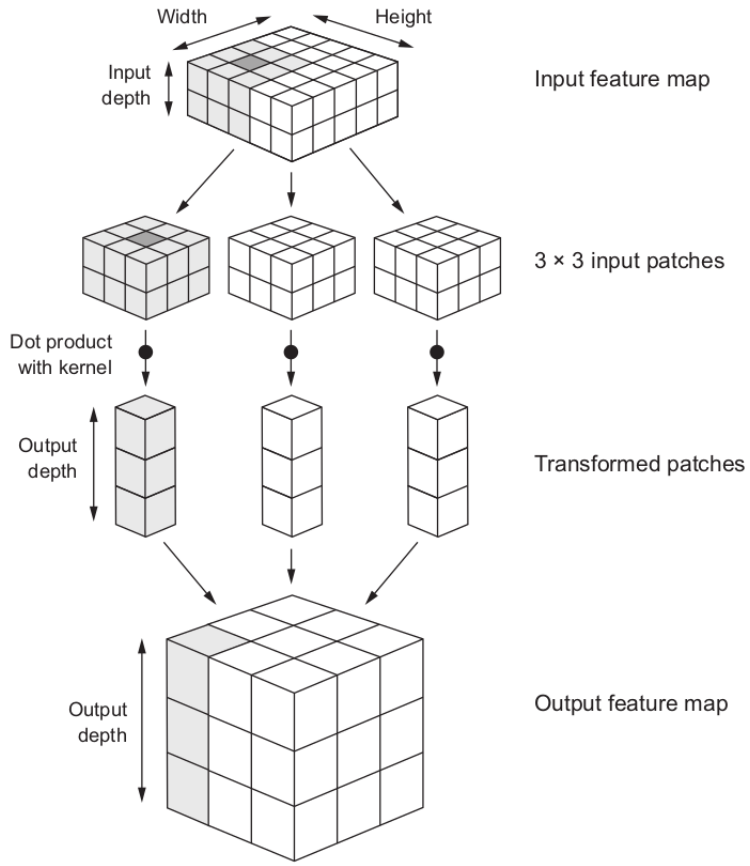

Figure 14 – **Scheme of a general spatial transformation of a convolutional layer.** A $(5 \times 5 \times 2)$ input image is processed by the convolutional operation using $(3 \times 3 \times 2)$ input patches (extracted by the kernels, of shape (3×3)). They extract transformed patches of size $(1 \times 1 \times 3)$, where 3 is related to the number of filters used in the operation, retulting in an output feature map of dimensions $(3 \times 3 \times 3)$. *Source:* Reference (179).

input matrix with a unit stride, they extract local patterns and features. Each kernel produces a transformed patch, resulting in vectors of dimension $(1 \times 1 \times 3)$, where 3 corresponds to the depth due to the number of filters. The final output feature map is constructed by combining all these transformed patches. It contains information about each portion of the original image corresponding to each filter used in the convolutional layer. The dimensions of this output feature map are $(3 \times 3 \times 3)$, where (3×3) represents the updated spatial dimensions, and the last 3 represents the depth (number of filters). This process demonstrates how convolutional layers effectively extract features from input images, allowing for hierarchical learning and representation of visual information.

Padding is also an important feature in convolutional layers, being often used to control the spatial dimensions of the output feature maps (179). When padding is applied, additional pixels are added around the borders of the input matrix before the convolution operation takes place. This ensures that the spatial dimensions of the output feature map remain the same as those of the input image, helping to preserve the information in the borders of the image. In

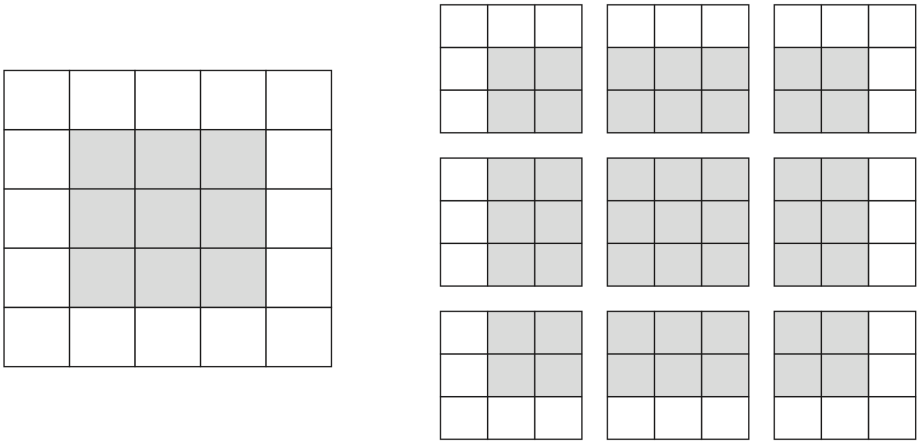

Figure 15 – **Example of padding operation.** An input image (on the left) of shape (3×3) is filled with an unit border, in order to process the *padding operation*. Defining an stride of 1 and a (3×3) kernel, the output feature map (on the right) has its shape of (3×3) (composed by the result of each kernel through the input image). *Source:* Reference (179).

Figure 15 we can visualize how padding works: by adding extra pixels (white pixels) around the borders of the input matrix (depicted as gray pixels), the convolutional operation results in a set of portions which, taken together, keep the original shape of the input image.

As mentioned previously, the convolutional layers are similar to the dense ones, having the addition of a bias and the application of an activation function. More specifically, after the convolutional operation, a bias term is added to each output feature map. Then, a nonlinear activation function is applied to the resulting feature maps. This transformation is done for each filter in the layer. Moreover, CNNs can be initialized with constant values, or sampled from predefined distributions for their initial weights.

Transposed convolutional layers, also known as *deconvolutional layers*, serve the opposite purpose of standard convolutional layers. While standard convolutional layers are used to reduce the spatial dimensions of input feature maps, transposed convolutional layers are employed to upsample, or to increase the spatial dimensions of the input. These layers achieve upsampling by applying a reverse convolution operation, where each pixel in the input feature map is expanded to a larger region in the output feature map. This expansion is controlled by the size of the kernels, the number of filters, and the stride used in the transposed convolutional operation. For example, consider a $(2 \times 2 \times 3)$ input matrix, a (2×2) kernel, 32 filters, and a (1×1) stride. After applying the transposed convolutional operation, the resulting feature map would have dimensions of $(3 \times 3 \times 32)$, where each pixel in the input matrix has been "increased in size" according to the dimensions of the kernels and strides.

3.3.2.1.2 Pooling Layers

Pooling layers play a crucial role in downsampling the size of input feature maps while

preserving important spatial information. These layers operate by selecting either the *maximum* or *average* value within each kernel-sized window, depending on whether it is a *max-pooling* or *average-pooling* layer. By specifying the kernel size and stride, pooling layers summarize the features present in local regions of the input data, effectively reducing the spatial dimensions of the feature maps. Additionally, pooling layers can aid in preventing overfitting by reducing the number of parameters and summarizing the most salient features of the input data. It is important to note that pooling layers preserve the depth size, which corresponds to the number of channels in the input feature maps. These layers are typically applied after convolutional layers in CNN architectures.

Conversely, *UpSampling layers* are used to increase the size of input matrices by repeating the values in the rows and columns. Similar to pooling layers, UpSampling layers allow for the manipulation of spatial dimensions in the feature maps, but in the opposite direction. By specifying the kernel size and stride, UpSampling layers enable the enlargement of feature maps, which can be useful for tasks such as image super-resolution or increasing the resolution of feature maps before further processing.

3.3.2.1.3 Dropout Layers

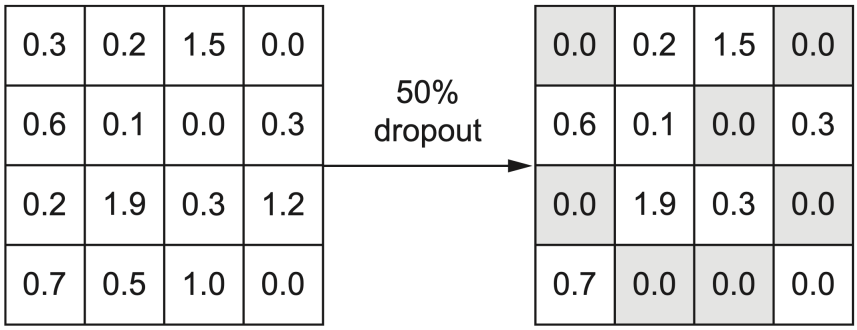

Figure 16 – **Dropout example of 50%.** The dropout operation preserves the shape of the input matrix by setting a designated fraction of pixels to 0 value. *Source:* Reference (179).

Dropout layers are a crucial tool for preventing overfitting in CNNs (179). They address this issue by randomly dropping a fraction of the output values during training. The *dropout rate* determines the fraction of output values that are set to zero in the output feature map. For example, a dropout rate of 50% means that half of the output values will be randomly set to zero (see Figure 16). Importantly, dropout layers preserve the spatial dimensions and number of channels of the input feature maps. This means that the height, width, and depth of the feature maps remain unchanged after the dropout operation. By retaining the spatial structure of the feature maps, dropout layers effectively regularize the network while preserving spatial information.

The flexibility of employing different flavors of layers to build CNNs allows them to excel in a wide range of tasks. For example, in the field of Astronomy, CNNs have been applied to tasks such as classification astronomical objects in images (207) and for detecting galaxy morphology (208). Moreover, CNNs have been employed in the analysis of hydrodynamical simulations by converting simulation data into structured fields, such as matrices containing information related to different components, with the goal of performing parameter estimation (209).

3.3.3 Image denoising techniques

The goal of image denoising is to recover a clean version of an image $\mathbf{x}$ from a noisy observation $\mathbf{y}$, where the noise $\boldsymbol{\nu}$ is typically modeled as Gaussian with zero mean μ and standard deviation σ. Mathematically, this relationship is described by:

$$\mathbf{y} = \mathbf{x} + \boldsymbol{\nu} \, . \tag{3.8}$$

An image denoiser should be able to produce a cleaned version $\hat{\mathbf{x}}$ that closely resembles the original clean image $\mathbf{x}$, by effectively reducing the noise present in the observed image $\mathbf{y}$. It is crucial for the denoiser to preserve the essential properties and features of the original image without introducing new artifacts or distortions. Various methods have been developed to tackle the problem of image denoising, ranging from traditional image filters tailored to specific types of noise, to more sophisticated ML techniques. For comprehensive reviews on image denoising techniques, we suggest References (210–212).

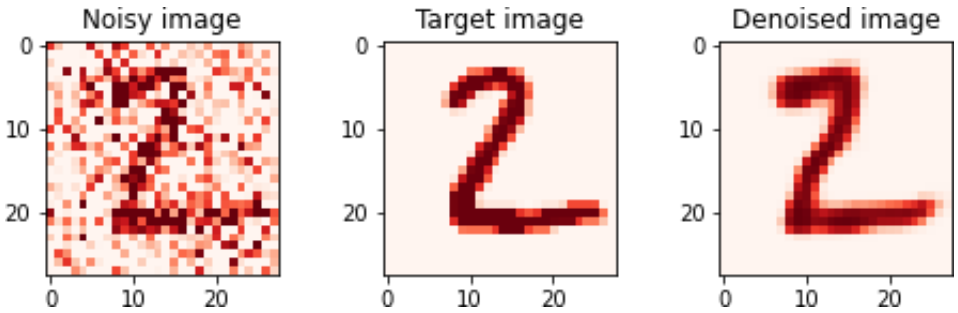

Figure 17 – **Example of image denoising applied to the digit 2, from the handwritten digits of MNIST.** The noisy image is showed on the left panel, the target image on the middle panel, and the denoised image is on the right panel.

Figure 17 illustrates the process of image denoising using a simple autoencoder, applied to an image of the digit 2 from the MNIST data set. The main idea is to observe that the algorithm takes the noisy image and, based on the target version, it is able to remove the noise from the peripheral regions of the image and enhance the true features. Depending on the application, the denoised image can be quite close to the target one.

Auto-encoders represent one of the main methods chosen in terms of performance for this task, mainly because they are purely data-driven, with no assumptions about the nature of

the noise (179, 211–213). For this reason, it was the method chosen in the work presented in Chapter 4. These methods take a pair of images: a noisy image $\mathbf{y}$ (input) and a clear image $\mathbf{x}$ (target). Given many such pairs, they learn to recognize what is signal, what is noise, and how to remove that noise, predicting images which are closer to the ones used as target – i.e., as close as possible to the ground truth images $\mathbf{x}$.

Usually, a basic auto-encoder has two parts: an encoder, followed by a decoder. First, the encoder takes an input image $\mathbf{y}$, of dimensions $d \times \bar{d}$, and maps it into a hidden representation $\mathbf{z}$, of dimensions $d' \times \bar{d}$, performing a mapping $\mathbf{z} = f_\Theta(\mathbf{y}) = f(\mathbf{W}\mathbf{y} + \mathbf{b})$, parameterized by $\Theta = \{\mathbf{W}, \mathbf{b}\}$. Here, $\mathbf{W}$ is a weight matrix and $\mathbf{b}$ a bias, both with dimensions $d' \times d$. Then, the decoder takes $\mathbf{z}$ and maps it back into $\hat{\mathbf{x}} = g_{\Theta'}(\mathbf{z}) = g(\mathbf{W}'\mathbf{z} + \mathbf{b}')$, using another weight matrix $\mathbf{W}'$ and another bias vector $\mathbf{b}'$. Hence, the reconstructed/denoised image $\hat{\mathbf{x}}$ has the same dimensions as the input/noisy image $\mathbf{y}$. In this way, the auto-encoder comprises a sequence of convolutional layers that are responsible for extracting features from the images, capturing the abstraction of their content, and then recovering the features at the end of the process. This is performed with the requirement that the loss function is minimized:

$$min_{\Theta,\Theta'} \left[\mathcal{L}\left(\hat{\mathbf{x}}, \mathbf{x}\right)\right] = F\left(\mathbf{y}, \Theta, \Theta'\right) \ , \tag{3.9}$$

where $F(\cdot)$ is the function learned by the CNN in order to remove the noise of the images, Θ and Θ' represent the set of parameters of the CNN, and $\mathcal{L}(\cdot)$ is the loss function (211, 214). The idea is that the loss measures the difference between the network predictions $\hat{\mathbf{x}}$ and the target images $\mathbf{x}$, such that the minimization of the loss function optimizes the function F that removes the noise.

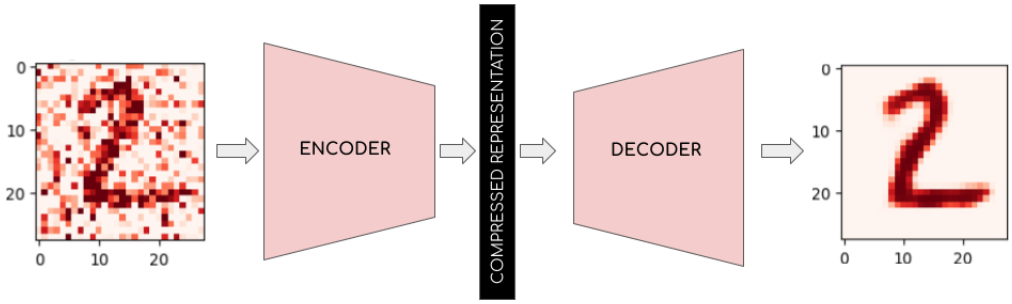

Figure 18 – **General scheme of a denoiser auto-encoder.** It shows a noisy image of the digit 2, which feeds the auto-encoder and produces a clean version of it. The encoder is depicted reducing the shape of the image, followed by the compressed representation and the decoder, responsible for decompressing the image, returning it to the input dimensions.

Auto-encoders are therefore built with: (i) the encoder, a sequence of 2D convolution layers (which can incorporate pooling and dropout layers as well), (ii) the compressed representation, consisting of a flattened layer followed by a dense layer and a reshaping layer (to obtain a compressed representation of the image, which goes into the *latent space*); and (iii) the decoder, a sequence of the 2D transposed convolutional layers (which can also include UpSampling and

dropout layers). A general scheme of an image denoising auto-encoder can be seen in Figure 18. The compressed representation is not mandatory: in those cases the implementations are called *fully convolutional and deconvolutional*, as they consist of a sequence of 2D convolutional layers followed by a sequence of 2D transposed convolutional layers. This was precisely the architecture implemented in the code used for the work presented in Chapter 4. The training is done by backpropagation, choosing the number of epochs to train, and finally obtaining the final set of weights (179, 200).

Image denoising techniques are often used for denoising and enhancing astronomical images – see, e.g., UNets (215). There are numerous other ML denoising techniques that are beyond the scope of this thesis, as we are solely employing this method to clean cosmological covariance matrices (see Chapter 4). However, interested readers may explore further this issue in References (210–212). In the work conducted in Chapter 4 we also employed another architecture called Residual Encoder-Decoder Network (REDNet) (213). REDNet follows the basic scheme of the auto-encoder presented here, symmetrically linking convolutional layers (belonging to the encoder) and de-convolutional layers (belonging to the decoder). However, it employs what is called by *residual* implementation because the information from a previous layer is added to the next one. Also, they do not consider the compressed representation. All the denoising methods presented in this thesis were implemented using the Keras library (200).

3.3.4 Graph Neural Networks

Graph Neural Networks (GNNs) constitute another category of neural networks, designed for processing structured and irregular data[6], which can be represented as *graphs*. Almost everything can be translated into a graph, which is one of the primary reasons behind the success of GNN applications in various domains such as chemistry (particularly in dealing with molecules), computer vision, natural language processing, and particle physics. Here, in Sections 6.1 and 6.2, we explore their capabilities for dealing with galaxy and halo catalogs. Specifically, GNNs can be used for tasks such as graph or node classification, link prediction, community detection using graph structures, prediction of global properties (graph embedding), or even graph generation (216–218). For comprehensive reviews on GNNs, readers can see References (216–220).

GNNs are distinguished by their ability to capture symmetries such as *permutational invariance* and *equivariance*[7], which make them even more attractive for science-related appli-

[6] Graphs can be *structured data* in the sense that they have well-defined relationships between entities represented as nodes and edges. On the other hand, graphs can also exhibit *irregular* characteristics. Graphs can vary widely in terms of size, connectivity, and topology. Some graphs may be densely connected, while others may be sparse. Additionally, the distribution of node degrees (i.e., the number of edges connected to a node) within a graph can be highly variable.

[7] A function $f(\mathbf{X})$ is *permutational invariant* if, for every permutation matrix $\mathbf{P}$, we have $f(\mathbf{PX}) = f(\mathbf{X})$. If this function acts as $f(\mathbf{PX}) = \mathbf{P}f(\mathbf{X})$, it is said to be *permutational equivariant*.

cations (219). Firstly, by construction, GNNs preserve the graph symmetries while remaining *permutational invariant*. This means, for instance, that global attributes do not depend on the node ordering. Secondly, their architecture is *permutational equivariant*, in the sense that reordering the input nodes produces the same permutation in the outputs. Thirdly, it is easy to incorporate graph attributes that account for other symmetries, as we will see in Section 6.1.1.3 (50).

3.3.4.1 Graphs

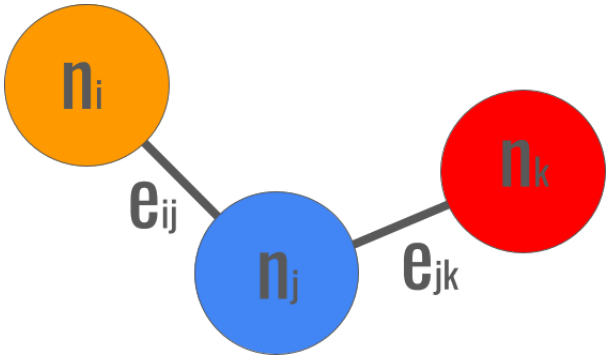

Figure 19 – **Example of a graph.** Here we show the nodes, along with each node attribute n_i, as well as the edges, along with their attributes e_{ij}. Note that the global attributes are not represented in this figure.

Graphs are mathematical structures defined by three main components: nodes, edges, and global properties. Each element of the graph can be described by a set of characteristics: $\mathbf{n}_i$ represents the properties or attributes of node i, $\mathbf{e}_{ij}$ represents the features of the edge between node i and j, and $\mathbf{g}$ contains the global properties of the graph. Therefore, a graph can be denoted as $\mathcal{G}(\mathbf{n}, \mathbf{e}, \mathbf{g})$ (217, 220, 221). A simple visualization of a graph can be seen in Figure 19.

Nodes i and j are considered *neighbors* if they are connected by an *edge*. We can represent all the connections in a graph using the *adjacency matrix* A_{ij}, which takes the value 1 if the pair is connected, and 0 if it is not. These connections can be either *direct* (with connections from the node i to the node j only) or *undirected* (with connections between nodes i and j, and vice versa). In the latter case, we say that we are considering reverse edges. A graph can also contains *loops*, representing edges that connect a node to itself. A *fully connected* or *complete* graph is one in which all nodes are connected by an unique edge. Otherwise, nodes can be connected by multiple edges and may even be "orphaned" (nodes that are not connected to any other node).

The *neighborhood* $\mathfrak{N}_i$ of node i includes every node j that shares an edge with it, such that

$$\mathfrak{N}_i = \{j | A_{ij} = 1\} . \tag{3.10}$$

This property will be used for defining the GNN architecture when updating the graph units (see Section 3.3.4.2.1).

In many ML applications (with the exception of those involving graph data sets, such as those described in Reference (222)), the use of GNNs necessitates the definition of graphs. This involves translating the data set at hand, often a tabular data set, into graphs. This process typically entails defining the data properties, whether relational or spatial, which determine the connections between objects (nodes), and subsequently defining node and edge attributes. In the context of Section 6.1, for example, galaxy catalogs are transformed into graphs, with galaxies serving as nodes along with their associated properties serving as node attributes. The edges between nodes are determined in terms of the galaxies' 3D spatial positions, with the possibility of additional properties being defined for the edges.

When considering a spatial property to establish connections between nodes, and to construct the neighborhoods of each node, various approaches can be employed (222). One such approach involves linking neighbors within a fixed radius (denoted as r_{link}) around each node. An alternative approach is to select the k nearest neighbors for each node. Both methods require the specification of an external parameter (either r_{link} or k) to establish connections between nodes, and the resulting graphs will vary depending on that parameter. However, different applications may favor one approach over the other. In the context of the applications discussed in Sections 6.1 and 6.2, we opted for the former approach – for more details, see Section 6.1.

3.3.4.2 Graph Neural Network Blocks

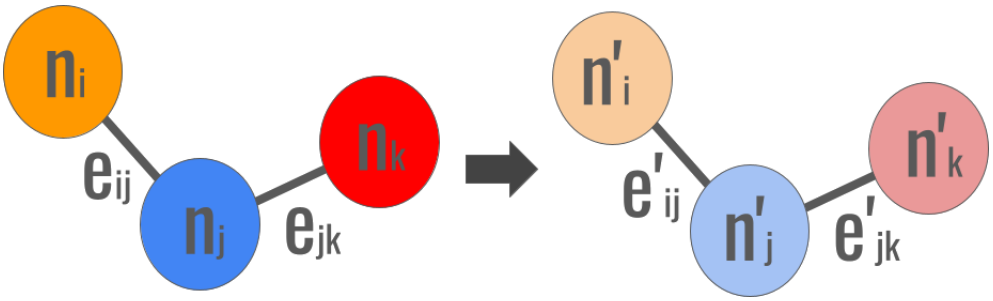

Figure 20 – **Scheme of an updated graph, after a GNN block, with their node and edge attributes updated.** The colors of the nodes are changed to represent this update, but the structure of the graph (the edge connections) is still the same.

GNNs are neural models designed to capture the relationships within graphs through a process known as *message passing scheme* among the graph components (217). This entails taking a graph as input and producing an updated graph as output, with various methods available to update the attribute values of the graph elements (nodes, edges, and global properties). In essence, the values of node, edge, and global attributes evolve throughout the process, while maintaining the connectivity among the nodes (i.e., the graph structure). We refer to the scheme depicted in Figure 20 in order to illustrate the scheme. This process of updating from one layer to another across different parts of the graph resembles the *passing of messages*

through the network, hence the name. Similarly to what is done in the case of CNNs, GNNs can be constructed using different layers or blocks, which we will explore further in this section.

3.3.4.2.1 Meta Layer

The meta layer (MetaLayer) serves as the foundational layer for constructing any GNN. It draws inspiration from Reference (220) and finds application in the PyTorch-Geometric library (222) for crafting other kinds of layers (e.g., graph convolutional and SAGE convolutional layers). Moreover, it constitutes the core layer utilized in the GNN architectures outlined in this thesis. This involves the user specifying what is referred to as the *node model*, *edge model*, and *global model*. These models are responsible for updating node, edge, and global attributes within the graph, according to the message passing scheme. Such updates are executed based on the values of the previous layer (or even the input graph values).

Since all these models depend on user specifications, and this model was employed in Section 6.1, we will now outline the specific choices made by us within that architecture. The layer $\ell + 1$ is updated using information from the layer ℓ according to:

- **Edge model:**

$$\mathbf{e}_{ij}^{(\ell+1)} = \mathcal{E}^{(\ell+1)}\left(\left[\mathbf{n}_i^{(\ell)}, \mathbf{n}_j^{(\ell)}, \mathbf{e}_{ij}^{(\ell)}\right]\right), \tag{3.11}$$

  where $\mathcal{E}^{(\ell+1)}$ is the *message function*, a differentiable function representing a MLP;

- **Node model:**

$$\mathbf{n}_i^{(\ell+1)} = \mathcal{N}^{(\ell+1)}\left(\left[\mathbf{n}_i^{(\ell)}, \bigoplus_{j\in\mathfrak{N}_i} \mathbf{e}_{ij}^{(\ell+1)}, \mathbf{g}\right]\right), \tag{3.12}$$

  where $\mathfrak{N}_i$ represents all neighbors of node i (see Equation 3.10), $\mathcal{N}^{(\ell+1)}$ is the *message function* (a MLP), and $\oplus$ is the *aggregator*, a *multi-pooling operation* responsible for concatenating several permutation-invariant operations:

$$\bigoplus_{j\in\mathfrak{N}_i} \mathbf{e}_{ij}^{(\ell+1)} = \left[\max_{j\in\mathfrak{N}_i} \mathbf{e}_{ij}^{(\ell+1)}, \sum_{j\in\mathfrak{N}_i} \mathbf{e}_{ij}^{(\ell+1)}, \frac{\sum_{j\in\mathfrak{N}_i} \mathbf{e}_{ij}^{(\ell+1)}}{\sum_{j\in\mathfrak{N}_i}}\right]. \tag{3.13}$$

The use of the multi-pooling operation in the equation above was made because it has been argued that several aggregators can enhance the expressiveness of GNNs (223) – but we can use only one of these operations (maximum value, sum, or average). Note that, in this example, we do not have a *global model*. However, this can be implemented in the same fashion as the other models, using the node, edge, and global attributes from the previous layers.

These layers are particularly useful when designing graphs that incorporate all their components, i.e., node, edge, and global properties, as they enable the updating of all these attributes, resulting in an updated version of the entire graph.

3.3.4.2.2 Graph Convolutional Layers

Graph Convolutional Layers are inspired by the work in the References (224,225). These layers are designed to perform the convolution operation, akin to matrix multiplication, on graphs, while updating node attributes using primarily node or edge information. We will discuss each one of them in detail below.

- **Graph Convolutional Network Layer**

  Graph Convolutional Network (or GCN) layers, as described in References (222, 225), are mathematically defined as

  $$\mathbf{n}_i^{\ell+1} = \sum_{j \in \mathfrak{N}_{(i)\cup i}} \frac{1}{\sqrt{\deg(i)} \cdot \sqrt{\deg(j)}} \cdot \left[\mathbf{W}^T \cdot \mathbf{n}_j^{(\ell)}\right] + \mathbf{b}, \tag{3.14}$$

  where $\mathbf{n}_i^{\ell}$ represents the node attributes at layer ℓ, $\mathbf{W}$ is a learnable weight matrix, $\deg(i) = |\mathfrak{N}_i|$ represents the degree of node i, $\mathbf{b}$ is an additive bias, and the sum is over the neighboring node features in the aggregator operation. The weight matrix is responsible for transforming the neighboring attributes, and the degrees of the nodes work as normalization factors.

- **Edge Convolutional Layer**

  Edge Convolutional (EdgeConv) layers, as described in References (222, 224), process graphs as follows

  $$\mathbf{n}_i^{(\ell+1)} = \max_{j \in \mathfrak{N}_j} \mathcal{H}^{(\ell+1)}\left(\left[\mathbf{n}_i^{\ell}, \mathbf{n}_j^{\ell} - \mathbf{n}_i^{\ell}\right]\right), \tag{3.15}$$

  where max is an aggregator operation, $\mathcal{H}^{(\ell+1)}$ represents a MLP, and $\mathbf{n}_i^{\ell}$ represents the node attributes at layer ℓ. Note that the node $\mathbf{n}_i^{(\ell+1)}$ has their values changed using its previous value $\mathbf{n}_i^{\ell}$, and the edge information is captured by the relative source node features $\mathbf{n}_j^{\ell} - \mathbf{n}_i^{\ell}$, for each edge (j, i).

Together, GCNs and EdgeConv layers can update the graph node attributes exclusively, in contrast to the MetaLayer discussed in Section 3.3.4.2.1, which updates both node and edge attributes. Therefore, they are particularly useful for GNNs designed to handle graphs with only node attributes.

3.3.4.2.3 SAGE Convolutional Layer

SAGE Convolutional layers, or GraphSAGE convolutional layers, are inspired by Reference (226). The main concept behind this layer is to update node information based on the

neighborhood of each node. The update for layer $\ell + 1$ is performed using information from layer ℓ as follows:

$$\mathbf{n}_i^{\ell+1} = \mathbf{W}_1 \mathbf{n}_i^{\ell} + \mathbf{W}_2 \cdot \text{mean}_{j \in \mathfrak{N}_i} \mathbf{n}_j^{\ell} \,, \tag{3.16}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are learnable weight matrices that can be viewed as learnable functions of the node values and their neighborhoods, respectively. Additionally, the mean aggregation function can be substituted with any other aggregation function (e.g., maximum, average, etc). Two alterations can be used to enhance the expressive power of the GraphSAGE layers, both related with the application of a nonlinear operation/activation function σ as:

- Before aggregation and conjunction with a linear operation (similar to MLPs) (226), as

$$\mathbf{n}_j^{\ell} \rightarrow \sigma^{\ell} \left(\mathbf{W}_3 \mathbf{n}_j^{\ell} + \mathbf{b} \right) . \tag{3.17}$$

  This includes an additional learnable weight matrix $\mathbf{W}_3$ and a bias vector $\mathbf{b}$ to the Equation 3.16.

- Applied to each layer (66):

$$\mathbf{n}_i^{\ell+1} \rightarrow \sigma \left(\mathbf{n}_i^{\ell+1} \right) . \tag{3.18}$$

Similar to convolutional layers, GraphSAGE layers are useful when applied to graphs where edge and global attributes are not considered, meaning only the node properties are updated using information from neighboring nodes.

In summary, the *message passing scheme* forms the core of GNNs, enabling them to handle structured and irregular data structures represented as graphs, which contain richer information compared to simple arrays. GNNs can be constructed using various building blocks, such as those discussed in this section or others. Many GNN architectures are composed solely of a sequence of MetaLayers (updating node, edge and global graph attributes), while others combine GCN, EdgeConv, and GraphSAGE layers, especially when focusing on updating node attributes exclusively. It is important to note that the MLPs within these layers must be accompanied by an activation function, and their architecture parameters, such as the number of neurons and layers. Optimizing these architectures involves selecting these hyperparameters as well. Likewise, the choice of the number of GNN blocks or layers is crucial for the overall performance of the model.

3.3.4.3 GNN variations

There are some other architectures that we have explored in the work of Section 6.1 to assess the network's importance for some of the galaxy properties employed, which can be seen as "variations" of GNNs. They are known as *deep sets* and *no initial node attributes*, and we will describe them here.

3.3.4.3.1 Deep sets

Deep sets are a type of neural network architecture designed to operate on *sets* of unordered data. These architectures are most similar to traditional MLPs. Their main difference is that they operate on sets with vectors of same dimensions in number of features and different number of objects, which are invariant to permutations. Similarly to GNNs, their operations are equivariant. In some sense, while compared to GNNs, they can be seen as an architecture made to deal with "graphs without edges", i.e., only nodes are present along with their node attributes (227). We stress that deep sets are not a type of GNNs.

Essentially, one of their implementation options is done employing MetaLayers, where node attributes are updated per layer with a **node model** defined as follows:

$$\mathbf{n}_i^{(\ell+1)} = \mathcal{N}^{(\ell+1)}\left(\mathbf{n}_i^{(\ell)}\right). \tag{3.19}$$

In the case of the test employed in Sections 6.1.2.1 and 6.1.6.3, we used this architecture to measure the importance of the distribution of the galaxy velocity field.

3.3.4.3.2 No initial node attributes

We can also build graphs initially without node attributes and with edge and global properties. Additionally, we can use the same MetaLayer model as presented in Section 3.3.4.2.1 to update the graph after having a first layer defined according to the models:

- **Edge model:**

$$\mathbf{e}_{ij}^{(1)} = \mathcal{E}^{(1)}\left(\mathbf{e}_{ij}^{(0)}\right), \tag{3.20}$$

- **Node model:**

$$\mathbf{n}_i^{(1)} = \mathcal{N}^{(1)}\left(\left[\bigoplus_{j\in\mathfrak{N}_i}\mathbf{e}_{ij}^{(1)}, \mathbf{g}^{(0)}\right]\right). \tag{3.21}$$

Then, after this first layer, we will end up with node attributes, which will be updated. This approach can be used while measuring the influence of node contributions to predictions. In the case of the tests employed in Sections 6.1.2.1 and 6.1.6.3, we used this architecture to measure the importance of the distribution of the galaxy clustering information.

GNNs have been extensively used in Cosmology for converting galaxy and halo catalogs into graphs, as demonstrated in References (50–52, 228, 229), and this will be further explored in Sections 6.1 and 6.2. They have also been employed for inferring halo masses (63, 67), speeding up semi-analytic models (66), and rediscovering Newton's law (197). The implementation of all the architectures presented in this work was done using PyTorch Geometric (222).

3.4 Probabilistic Methods

Through all these previous sections, while dealing with a regression problems, we have primarily focused on ML methods designed to provide *point estimations* or *single-value predictions*. These methods are capable of making inferences by obtaining a single-value prediction y – in the case of an N-dimensional array, they provide single-value predictions for each entry in the array. Additionally, they are often referred to as maximum likelihood estimators, as they aim to predict point values while minimizing the loss function (see Equation 3.6).

However, there are methods to offer a different approach by estimating moments of a probability distribution, or they can even estimate full probability distribution functions[8]. In the subsequent discussion we will explore Moment Neural Networks, which are used to predict the first and second moments of the posterior distribution (μ and σ), as well as methods for predicting full probability density distributions, such as converting a regression problem into a classification one, using a method called *NNClass* (69).

3.4.1 Moment Neural Networks

Moment Neural Networks (MNNs) are designed to address the challenge known as the "curse of dimensionality" when dealing with high-dimensional probability density estimators (232). This concept refers to the increasing computational complexity associated with obtaining full probability density distributions, which grows with the number of dimensions in the problem. Since in many inference problems it may not be necessary to obtain the complete probability density distribution, we can achieve our goals by extracting only certain moments of that distribution. MNNs are particularly useful in this context, as they enable us to approximate the desired distribution in terms of its moments. This approach not only reduces computational expenses, but also facilitates the prediction of higher-dimensional properties. Even when focusing on a single property, such as estimating the density parameter Ω_m, MNNs can provide valuable insights, as demonstrated in Sections 6.1 and 6.2.

For instance, given data $\mathcal{D}$, and the goal of predicting the marginal posterior mean μ and standard deviation σ without making any assumption about the posterior, the task can be formulated as follows:

$$\mathbf{y}(\mathcal{D}) = [\mu(\mathcal{D}), \sigma(\mathcal{D})], \tag{3.22}$$

[8] There are other methods used with this same purpose, such as Bayesian Networks (230) or the generative Normalizing Flows (231). These other methods can achieve similar or better results, depending on the application problem. Describing these methods, however, is outside the scope of the present thesis, because they have not been used.

where

$$\mu(\mathcal{D}) = \int_y dy\ y\ p(y|\mathcal{D}), \tag{3.23}$$

$$\sigma^2(\mathcal{D}) = \int_y dy\ (y-\mu)^2\ p(y|\mathcal{D})\,. \tag{3.24}$$

Here, $p(y|\mathcal{D})$ represents the marginal posterior distribution. To train a MNN for this task, a specific loss function can be utilized, as described in Reference (232):

$$\mathcal{L} = \log\left[\sum_{j\in\text{batch}} (y_j-\mu_j)^2\right] + \log\left\{\sum_{j\in\text{batch}} \left[(y_j-\mu_j)^2 - \sigma_j^2\right]^2\right\}, \tag{3.25}$$

where j represents the samples in a given batch. By taking the logarithm of each term, the loss function effectively rescales both terms to the same order of magnitude, ensuring equal weight is given to predicting both the first and second moments (209).

MNNs have found applications being associated to any NN architecture (since direct used in MLPs to GNNs) in parameter estimation tasks (42, 48, 70) and in halo mass estimation (63, 67), for example. Their usage will be presented in detail in Sections 6.1 and 6.2, along with a GNN architecture.

3.4.2 Regression to Classification: NNClass

In this section we explore a technique for converting a regression problem into a classification one using MLPs (here coined as neural networks or NNs), which we refer to as NNClass (69). The objective of this method is to introduce uncertainties in the output predictions by transforming the task from predicting single values to estimating probability density distributions.

The process starts with defining K classes, by partitioning the predicted property y into K intervals (or bins). During training, the true values are categorized into appropriate bins based on their range. By employing a SoftMax activation function in the last layer of the network and using Categorical Cross Entropy as the loss function, similar to typical classifications tasks, the model assigns a score to each class (bin). The scores are normalized such that they sum up to one, providing a probabilistic interpretation of the output (even if the scores are not true probabilities).

This approach has been widely utilized in various domains, such as photometric redshift estimation (233–235). It was also employed in the research that is presented in Section 5.4.

3.5 Other Machine Learning Tools

In this section we explore some ML approaches that are not traditional ML methods, but that are sometimes employed to address some of the challenges commonly encountered in ML applications. In Section 3.5.1 we delve into a technique for handling imbalanced data

sets, particularly for regression problems with typical tabular data. Following that, in Section 3.5.2 we explore strategies for scenarios where multiple algorithms are available to solve the same problem, but selecting the best one is not straightforward. Then, in Section 3.5.3 we explain feature importance analysis, as a way to measure the influence of input features in predicting target variables. Finally, in Section 3.5.4 we present a method for navigating the hyperparameter space in order to identify the optimal set of hyperparameters for a specific problem.

3.5.1 The problem of Imbalanced Data Sets

In ML, *imbalanced data sets* can often be challenging. In these data sets, regions of the data space that are relatively underrepresented may carry equal or even higher importance within the scientific context of the analysis compared with the majority of the distribution. As a result, predicting these data points becomes more challenging for ML models, which typically end up focusing their learning in the regions around the peaks of the distribution in parameter space. In regression problems, there are two primary approaches to address this issue: pre-processing and model processing techniques.

Pre-processing techniques involve manipulating the data set before training the model. They often involve applying *over-sampling* and/or *under-sampling* techniques to increase or reduce the amount of data in specific regions of the distribution.

Alternatively, the importance of these regions can be weighted, which is the approach followed by the second set of techniques. One such method is the *Synthetic Minority Over-Sampling Technique for Regression with Gaussian Noise* (SMOGN), which primarily functions as an over-sampling technique (236). Specifically, SMOGN operates by combining random under-sampling with two over-sampling techniques: SmoteR (an adaption for regression of SMOTE[9] (237)) and Gaussian noise. The algorithm begins by binning the data based on a given target variable, dividing the resulting distribution into "rare" and "normal" bins. Rare bins are augmented, while normal bins are under-sampled. During augmentation, an object within a rare bin is selected, and its k nearest neighbors are identified using Euclidean distances. A neighbor is then randomly chosen, and if it is close to the the initial object (based on a predefined threshold, i.e., half the median distance of all k neighbors), a new object is generated by interpolating between them. This process ensures a "safe distance" (following the nomenclature of Reference (238)) from the initial object to its neighbors, preventing outliers from affecting the sampling. Gaussian noise is added during this process to create new objects closer to the original sample. Under-sampling follows a similar approach, randomly removing objects from the original bins.

[9] *Synthetic Minority Over-sampling Technique* (SMOTE) is a data augmentation method that improves imbalanced classification data sets. It works by over-sampling some minority class and under-sampling the majority class, using the kNN method. The main difference between SMOTE and SMOGN is that the first is build to predict discrete results, while the second is able to provide continuous predictions.

The result is a distribution with an increased number of objects in previously underrepresented regions and a decreased number of objects around the bulk of the distribution.

In the work presented in this thesis, the SMOGN library was utilized (239).

3.5.2 Combining different methods

One approach for combining many ML methods used to address the same problem with the same data set, based on their individual performance scores, is to utilize *ensemble* algorithms. We have already seen ensemble methods in the context of tree methods (see Section 3.2.2). These algorithms leverage the outputs of multiple methods to enhance the predictions made by each individual method.

One such ensemble method is *stacking regression*, which involves forming combinations of predictions from various predictors (240). Essentially, this technique aggregates predictions from diverse "weaker learners" in parallel, treating them as features and outputs for a more accurate single prediction by blending or meta-learning the model (241). By integrating predictions from different models, each with its own strengths and weaknesses, stacking regression can yield superior predictions with reduced variance compared to using a single model alone. This approach helps to mitigate overfitting, enhance model robustness, and minimize potentially inflated model performance scores.

In the work presented in Section 5.3, we combine the predictions for four individual ML methods using the LinearRegression module from SkLearn (182).

3.5.3 Feature Importance

Feature importance (FI) refers to the relative contribution or influence of input features in predicting the target variables. Understanding feature importance is crucial for interpreting the behavior of a ML model, identifying which features are most informative, and gaining insights into the underlying relationships in the data. Hopefully, a physical interpretation of the success of the model.

There are various techniques to measure feature importance. For instance, we already mentioned that tree methods are straightforward and intuitive to visualize and interpret. Because of these characteristics, they are naturally used to perform FI. In the most general case, a DT, the FI can be determined by examining how much each feature increases information gain when it is used to split nodes in the tree. Features that lead to significant gains in information are considered more important. The importance of a feature can be computed by summing the gains in information over all nodes where the feature is used for splitting, weighted by the number of samples in each node. The other tree methods follow similar principles.

One common method, which is not restricted to a category of algorithms (tree methods) is the *permutational feature importance*. This method basically measures the relative decrease

in a particular score (in the case of regression problems, the MSE for example), measured for different runs of the model after randomly removing one feature at a time (186). This can be done removing one feature at time from the data set and re-training the model, checking the score. However, this can be computationally intensive, especially considering a high-dimensional feature spaces. To avoid re-training the estimator we can remove a feature only from the test part of the data set, and compute score without using this feature. Because the model was trained considering a specific dimension of features, what is done is to replace the feature for random noise. The random noise is obtained shuffling values of the feature in question, i.e. using other examples' feature values while keeping the values of all other features unchanged. This effectively destroys any meaningful relationship between the shuffled feature and the target variable. Therefore, the performance is evaluated and compared to the performance in the usual test set. A larger drop in performance suggests that the feature is more important, as the model relies heavily on it for making prediction. In this thesis we have used Eli5 to perform this task.

3.5.4 Hyperparameter search

ML methods aim at constructing models that are able to make the best possible predictions given some data set. But many common learning algorithms feature a set of *hyperparameters* that must be determined (fixed) before training even begins. These parameters control aspects of the learning process itself, rather than being learned from the data. Examples of hyperparameters include the learning rate in gradient descent, the number of layers and neurons in NNs, the kernel size in CNNs, and many others.

The selection of hyperparameters can significantly influence the performance of the resulting model, convergence speed, and generalization ability, but finding optimal values can be challenging (242). Typically, this process involves manually adjusting these parameters and observing their effects on various metrics or scores. However, this approach can be difficult because changing one parameter at a time may not accurately reflect how the entire set of parameters behaves when tested together.

To address this challenge, *hyperparameter search*, also known as *hyperparameter optimization* or *tuning*, involves exploring different combinations of hyperparameter values to find the combination that results in the best performance of the model on a validation data set. This process is typically performed using a search algorithm, such as *grid search*, *random search*, or *Bayesian optimization*. Grid search involves specifying a grid of hyperparameter values and evaluating the model's performance for each combination of values in the grid. Random search samples hyperparameter values randomly from predefined ranges and evaluates the model's performance for each sampled combination. Bayesian optimization models the objective function (e.g., validation error) and uses probabilistic models to guide the search towards promising regions of the hyperparameter space.

Various frameworks have been developed for hyperparameter search, including Optuna (243), which is employed throughout this thesis. Optuna provides an efficient implementation of both searching and pruning strategies, and it is easy to setup with many ML methods and libraries. Essentially, the user specifies the hyperparameters to be searched and their ranges, as well as an early-stopping scheme that identifies the best set of hyperparameters based on minimizing the validation error. Optuna then samples the hyperparameter space using Bayesian optimization with Tree Parzen Estimator (TPE). This method can handle tree-structured search spaces, including conditional parameters, and utilizes Parzen estimators (also known as kernel density estimators, or KDE). In each iteration, TPE calculates the parameters and selects the configuration with the best acquisition function value based on samples from the KDE. This process continues until the early-stopping criteria are met. We suggest the References (244, 245) for more details regarding this method.

4 IMPROVING COSMOLOGICAL COVARIANCE MATRICES

Cosmological covariance matrices play a crucial role in parameter inference by propagating uncertainties from data to model parameters. However, accurately estimating these matrices for large data vectors requires a substantial number of observations or costly simulations, which may not always be feasible (see Sections 2.3.5 and 2.3.5.1). In this work, we propose a ML approach to address this challenge specifically in the context of covariance matrices used in the study of large-scale structure.

Using only a small amount of data, composed of matrices built from samples of 50 to 200 halo power spectra, we demonstrate the capability to produce significantly improved covariance matrices. These matrices closely resemble those constructed from much larger samples, comprising thousands of spectra. To achieve this, we train CNNs to denoise the covariance matrices, leveraging methods outlined in Sections 3.3.2 and 3.3.3. During training, we utilize a data set consisting entirely of spectra extracted from simple and cost-effective halo simulations (mocks).

Our results indicate that the denoising method effectively removes noise from the covariance matrices of the inexpensive simulations. More importantly, it also successfully denoises the covariance matrices of halo power spectra obtained from N-body simulations. Through various metrics, we compare the denoised matrices with the noisy sample covariance matrices, consistently finding significant improvements in the denoised matrices without any spurious artifacts.

By employing the Wishart distribution we demonstrate that the denoiser's output can be likened to an effective sample augmentation in the input matrices. Additionally, we show that using the denoised covariance matrices enables the recovery of cosmological parameters with nearly the same accuracy as when using covariance matrices constructed from larger samples. Notably, we observe a significant reduction in bias in the Hubble parameter H_0 after applying the denoiser.

The work presented in this chapter is related to the background seen in Sections 2.3.5 and 2.3.5.1 for parameter estimation using covariance matrices, and in Sections 3.3.2 and 3.3.3 for the ML methodology that was employed here. Additionally, the main achievements related to this work are associated with the publications of References (40, 246, 247).

Future prospects from these efforts are predominantly in the application of image denoising technique on real data. In that context, the approximated method already existing for the fiducial Cosmology chosen for the survey can be used to train the machinery, and then applied on covariance matrices from N-body simulations. These matrices can incorporate the survey footprint, selection functions, redshift errors, redshift-space distortions and higher-order

statistics.

4.1 Motivation

The culmination of cosmological data analysis is parameter inference, which plays a crucial role in constraining our theoretical models. Covariance matrices serve as the bridge between theory and data, as they quantify the expected fluctuations in our measurements based on both the underlying physical phenomena and the conditions under which the data sets are collected.

In the context of galaxy surveys, the physical phenomena encompass the Gaussian random nature of the initial density fluctuations, while the observational conditions encompass factors such as the total volume of the survey, the shape of that volume (referred to as the mask), the mean density of detected galaxies, and potentially other real-life variables. While analytical or semi-analytical methods may offer a reasonable initial approximation for the covariance matrix, both the physical models and the observational conditions are often best represented through simulations. By utilizing a set of independent simulations and a selection of summary statistics for analysis, we can compute the sample covariance. This sample covariance should accurately reproduce the statistical errors of these statistics, thereby providing constraints on the underlying physical models. However, it is crucial that sample covariance matrices are both precise and accurate (248). This necessitates not only well-designed models for the physical phenomena and data acquisition but also requires large samples. Without a sufficient sample size, there is a risk of biasing parameter estimation (37, 249).

However, the number of simulations necessary to properly characterize these effects, and to allow for unbiased estimation of the parameters, is often very large, which represents a daunting computational challenge (250), especially when it is important to properly model the nonlinear scales (251). Using the power spectrum as our summary statistics, the sample size (the number of simulations) that is typically required to fulfill those needs is around $N_s \sim n_k^2$, where n_k is the number of k bins (bandpowers), and this number can grow even more with different tracers of the large-scale structure and all the resulting auto- and cross-spectra. This number is now under pressure from two sides: on one hand, in order to test the physical phenomena of interest we need to increase the dynamic range of our surveys to both larger and smaller scales, but without losing resolution–and that means more bandpowers. On the other hand, astrophysical surveys are increasingly able, either by themselves or in combination, to map the universe with much greater completeness by detecting multiple tracers of the large-scale structure. Therefore, it is of paramount importance to optimize methods that can estimate efficiently, and with greater precision, these cosmological covariance matrices.

Several efforts have already been made with the goal of obtaining precise covariance matrices using smaller samples – for a review see Reference (37). This problem does not exist if we employ analytical approximations that try to codify the impact on nonlinear clustering (252).

Analytical models can also help to enhance covariance matrices, e.g., by using χ^2 distributions from simulations, leading to a reduction in the number of realizations needed to achieve a certain threshold in accuracy (253). In the absence of a large sample, a commonly used technique is the Jackknife method (36), which relies on sub-sampling an original data set. Another possible direction is data compression, which actually means reducing the size of the data vector by maximizing the Fisher information, which can be done both in the context of parameter-independent covariances (254), as well as parameter-dependent ones (250). A closely related method relies on reducing the dimensionality of the observables (the parameters), which then allows for a less noisy estimation of covariance matrices given the same sample size (255). It is also possible to resample some specific modes or parts of the data vector (256).

Last but not least, we have approximate numerical methods (as we have seen in Section 2.5.1.9), which in the context of large-scale structure are provided by PTHalos (139), EZMocks (140), PINOCCHIO (141), PATCHY (142), HALOGEN (143), LogNormal (144), ICE-COLA (145), ExSHalos (146), BAM (147, 148) and many other techniques, which all attempt at generating halo catalogs using semi-analytical approximations or by emulating much more expensive N-body simulations. However, even if the results seem compatible with the numerical simulations, as shown in Reference (149), covariance matrices derived from these approximation schemes can lead to statistical deviations of up to $\sim 5\%$ for the bias in the estimation of cosmological and nuisance parameters, and of around $\sim 10\%$ for the volumes in parameter space, when compared with the true (N-body) estimations, which may fall short of the accuracy required for precision cosmology from future surveys. Nevertheless, it is possible to use mocks in order to reduce the number of simulations needed for characterizing the statistics of the matter power spectrum or bispectrum, by exploiting the correlations between the mocks and N-body simulations–see, e.g. CARPoll (257, 258).

From a different perspective, ML techniques offer alternative solutions to some of these challenges. There are efforts trying to speed-up the process of producing high-resolution N-body simulations, by starting from lower-resolution ones, and letting the ML fill in the detailed structures on small scales (195, 259, 260). In References (261–263) the authors try to emulate the full nonlinear evolution of N-body simulations by inputting only approximate simulations of these.

In this work we propose a new approach, that employs CNNs, more specifically image denoising techniques, as a tool to enhance sample cosmological covariance matrices that are based on a small number of high-resolution simulations. As we will show, the final covariance matrices (after denoising) become as precise and accurate as the ones obtained with a much higher number of high-resolution simulations. The idea is to train the ML method using data coming from halo mock generators, and then to apply that machinery to improve ("denoise") a sample covariance matrix that was constructed using a small sample of very accurate, high-fidelity N-body simulations. In practice, we train the ML denoiser using covariance matrices for

power spectra from halo mocks produced by ExSHalos (146), and then we apply the denoiser to covariance matrices produced using matching halo catalogs extracted from the Quijote suite of N-body simulations (112). This process shows, first, the robustness power of generalization of the method, which is able to improve covariance matrices from a set of simulations that the ML has never seen before. And second, that the ML denoiser can generate cosmological covariance matrices that are in all respects equivalent to those produced from thousands of N-body simulations.

4.2 Halo catalogs

In this work we have used two different halo catalogs: the ones obtained from a halo mock generator, ExSHalos, are the "cheap" simulations, and the ones from the Quijote suite are the "high-fidelity" N-body simulations. In this section we briefly explain some of their main features, and our method for matching them so there is greater compatibility between the data sets.

4.2.1 ExSHalos

We have already seen details about ExSHalos in Section 2.5.1.9. In order to utilize this code into this work we used the input linear power spectrum from CAMB (111). Also, we have chosen the second input, the threshold density for halo formation in linear theory, as the constant barrier and used LPT to second order. The cosmology was chosen according to the standard one in the Quijote suite, as well as the size of the box, of volume $(1,000\,\mathrm{Mpc}/h)^3$, for which we used cubic cells of $1(\mathrm{Mpc}/h)^3$ volume at a fixed redshift $z = 0$. In total, we have produced $30,000$ mock catalogs (hereafter, $N_{max} = 30,000$ for ExSHalos). It should be noticed that not all the ML models used in this work needed to use this amount.

4.2.2 Quijote

The Quijote suite (112) was also quickly described in the beginning of Section 2.5.1. Additionally, we can say that the initial conditions were generated at redshift $z = 127$, using an input matter power spectrum and a matter transfer function computed with the help of CAMB, and were evolved up to $z = 0$. All the Quijote simulations have a volume of $(1,000\,\mathrm{Mpc}/h)^3$, with 512^3 cold DM particles. The suite has simulations for a range of cosmological models, but the main (standard) fiducial cosmology follows the Planck best-fit model (14): $\Omega_m = 0.3175$, $\Omega_b = 0.049$, $h = 0.6711$, $n_s = 0.9624$, $\sigma_8 = 0.834$, $M_\nu = 0.0eV$ and $\omega = -1$.

We have downloaded $15,000$ halo catalogs, which is the maximum number of Quijote simulations for the main fiducial Cosmology (hence, $N_{max} = 15,000$ for Quijote), all at $z = 0$, with halos identified using the Friend-of-Friends (FoF) algorithm (264) with linking length parameter $b = 0.2$. Those catalogs were downloaded using the Globus command line

interface–however, we stress that this larger sample was only used to test and validate our ML method: all the training was performed using the ExSHalos data set.

4.2.3 The match between the catalogs

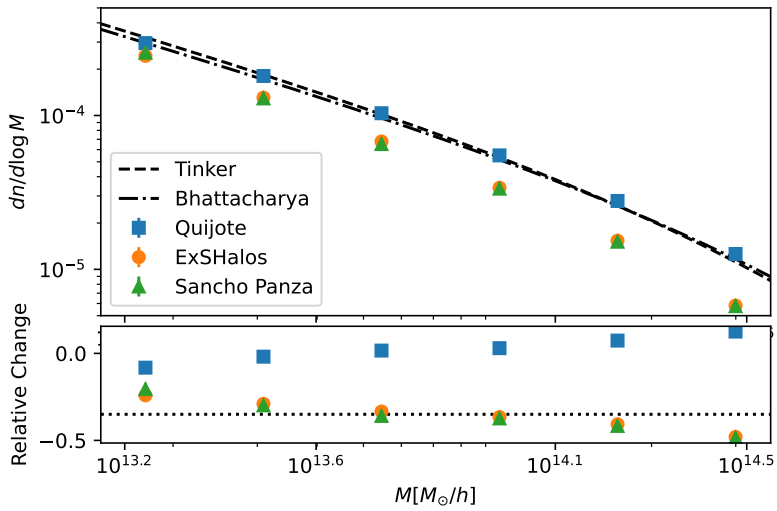

Figure 21 – **Halo mass functions comparison.** Quijote, ExSHalos, and Sancho Panza simulations compared to the phenomenological halo mass functions of Tinker (101) and Bhattacharya (102). The lower panel shows the relative change of the simulations with respect to the Tinker halo mass function. *Source:* Reference (40).

In order to match the two different halo catalogs we ensured that both Quijote and ExSHalos provided a similar numbers of halos, with approximately the same values for the halo bias. We have analyzed the halo mass functions of both catalogs considering halos in the mass range $M \in [10^{13.12}, 10^{14.6}]M_\odot/h$, in intervals of $\Delta \log_{10} M \simeq 0.25 \log_{10}(M_\odot/h)$, which corresponds to 6 bins of halo mass. We then compared the resulting halo mass functions from the catalogs with fits from Tinker (101) (using $\Delta = 200$ and $b = 0.2$ to compare with Quijote halos) and Bhattacharya (102), which were computed with the help of the Colossus library (265).

The results of this comparison are shown in Figure 21. From that figure it is clear that ExSHalos do not show a perfect agreement with the phenomenological fitting functions: the lower panel shows a relative difference with respect to Tinker's fitting function that hovers around a deficit of $\sim 35\%$ (which is represented by the dotted black line in the lower plot). When compared to halos from real N-body simulations (Quijote), the results agree with both Tinker and Bhattacharya halo mass functions on almost all scales, deviating only $\sim 10\%$ at the lower mass end. Even if the halos in ExSHalos and Quijote had the same bias, the different abundances would have an impact on shot noise and in the covariance of the power spectra. Hence, in order to improve the match between ExSHalos and Quijote, we randomly removed halos from each mass bin in the Quijote catalogs in such a way that their final abundances

match the same halo bins in ExSHalos–in that respect, see also Reference (149). We refer to the resulting "clipped" Quijote catalogs as the Sancho Panza sample.

Not only the halo mass was matched, but we have tested the halo bias, for these same mass bins, to compare the results between ExSHalos and Quijote/Sancho Panza maps and we have observed that they were similar for each mass bins. The detailed comparison is presented in Section 4.3.2.

4.3 The data set of covariance matrices

In this section we describe how we simulate a survey with a non-trivial mask, and show the effects of that mask on the power spectrum of the tracers in the two simulations. We also make a preliminary check that the bias in the two samples is nearly the same, as well as the dynamical range of scales that we are able to analyze. At the end of the section we describe the construction of the samples and the computation of the covariance matrices.

4.3.1 The power spectrum

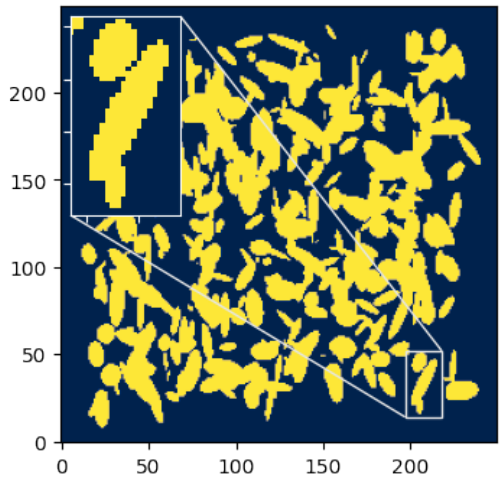

Figure 22 – **Slice of the mask of random ellipsoids.** The inset shows a zoom in on three ellipsoids, for a better visualization. *Source:* Reference (40).

For the sake of simplicity, in this work we chose as tracers the halos belonging to the first mass bin discussed in Section 4.2.3, i.e., halos with masses between $M \in [10^{13.12}, 10^{13.37}]M_\odot/h$, with a mean mass $\langle M \rangle = 10^{13.245} M_\odot/h$, both in ExSHalos and in Sancho Panza samples[1]. This choice minimizes the differences in the halo mass functions of the two samples, which are also closer to the mass functions by Tinker and Bhattacharya for that mass bin.

[1] Note that the results presented here do not depend significantly on our choice of tracer, especially for the parameter estimation. Nevertheless, different tracers have different cosmological covariance matrices: e.g., shot-noise may affect the diagonal and off-diagonal terms of the covariance matrices in different ways.

The power spectrum is our chosen summary statistics, so that for each simulation the data vector corresponds to a set of $P(k)$, and that is what we use to build the covariance matrices. In order to simulate real-life effects and power spectra which are closer to the ones measured in realistic cosmological surveys, we have *masked*[2] the halo maps in a way that attempts at emulating a mask covering some regions of the sky: this mask reflects the survey's footprint as well as bright stars, cloudy nights, regions with poor seeing, etc (266). The main point of using a mask, in the context of this work, is to induce non-trivial correlations between different spectral modes, in a way that resembles real surveys.

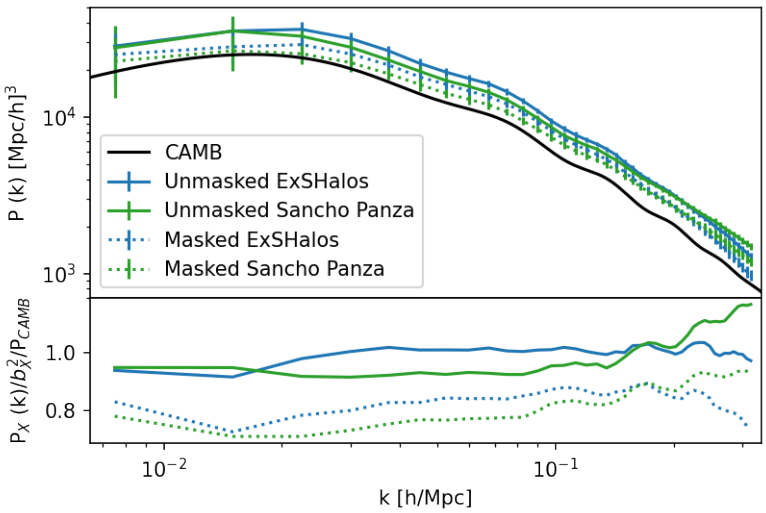

Figure 23 – **Power spectra comparison.** We compare the power spectra of ExSHalos (blue lines) and Sancho Panza (green), presenting their masked (dotted) and unmasked (solid) versions. The linear matter power spectrum is also shown as the solid black line for comparison. *Source:* Reference (40).

Our mask was built using a single realization of randomly placed ellipsoids with random sizes and orientations, in such a way that regions outside those ellipsoids were masked out (i.e., regions outside the mask are assigned weight zero, while regions inside it have weight 1). The mask occupies approximately $\sim 50\%$ of the total box volume, and we estimated the spectrum on a grid of the entire box, with cells of $4\,h^{-1}$ Mpc on a side. Figure 22 shows a slice of our mask, with the sizes and shapes of the ellipsoids as well as shapes of the mask edges with respect to the cell size that we used in our simulations.

We computed the spectra of both the masked and unmasked catalogs, which are shown (in terms of the mean for 100 maps) in Figure 23. It can be seen that the shapes of the power spectra of the ExSHalos and Sancho Panza samples are similar on scales larger than $k \lesssim 0.15\,h\,\mathrm{Mpc}^{-1}$. On smaller scales, although the shot noise for both samples are identical, the differences start to become more evident, and are due to the approximate treatment of structure formation in halo mocks such as ExSHalos. The effect of the mask can also be seen from this plot, in terms of a suppression of the clustering amplitude on all but the smallest scales.

[2] We will see another approach to mask effects in Section 6.3, while masking galaxy catalogs.

4.3.2 The bias

In order to check the similarity between the ExSHalos and Sancho Panza halo catalogs we also computed the bias of the halos in our chosen mass bin. We have used the bias definition from Tinker (103), the same used in Section 2.3.4, see Equation 2.3.4. The bias was computed for 100 ExSHalos and 100 Sancho Panza catalogs, and averaged over scales in the range $k \in [0.015, 0.2625]\ h\ \mathrm{Mpc}^{-1}$, where the small-scale cut-off corresponds to the value of k for which shot noise becomes 80% of the power spectrum (267).

Table 1 – **Bias for the halo catalogs.** Comparison of ExSHalos, Sancho Panza, and fit bias following Tinker and Bhattacharya (102, 103).

Catalog	Bias	Tinker	Bhattacharya
ExSHalos	1.245 ± 0.037	1.208	1.096
Sancho Panza	1.223 ± 0.057		

The results of this comparison are presented in Table 1, where besides the mean and standard deviation of the halo bias for our maps we also show the expected bias obtained with the Tinker (103) and Bhattacharya (102) fits. We should stress that, when we perform our cosmological parameter inference, we treat the bias as a nuisance parameter–see Section 4.5.

4.3.3 The covariance matrices

All the cosmological covariance matrices in this work were built by computing the sample covariance for the power spectra in the data vector according to Equation 2.34 (see Section 2.3.5.1). We should regard any sample covariance as a random matrix: given two different samples of N data vectors, the two sample covariances will be different by an amount that reflects the level of statistical fluctuations in the two samples. Larger samples are naturally less subject to those fluctuations, but they require more simulations. Our goal here is to show that ML techniques are able to identify (and correct) at least part of those patterns of statistical fluctuations, even if they are trained using simplified simulations.

The data set of sample covariance matrices used to train the ML suite was built using only the spectra from ExSHalos catalogs, whereas the data set used for the final validation of the predictive power of the suite was built using spectra from Sancho Panza. In order to train, validate and test our ML denoiser we used samples of different sizes. First, we constructed the *input* covariance matrices, which correspond to small sample sizes (n spectra), and are the ones we would like to enhance with our denoiser. Next, we computed covariance matrices with larger sample sizes (N spectra, the *target* matrices), which serve to teach the ML denoiser about how to clean the noise of the input matrices by performing the task $n \to N$. Finally, we compare the cleaned version of the input matrices with the *best* possible covariance matrix,

which is computed using the maximum sample size (N_{max} spectra), corresponding to the total entire set of catalogs.

We have used target sample covariance matrices with $N = 1,000$ spectra, and input matrices with sample sizes in the interval $n \in [50, 250]$, in increments of $\Delta n = 25$. These choices were informed both by the typical number of mocks used in the estimation of cosmological covariance matrices, and by our goal to test what is the minimum number of input spectra (n) that results in denoised matrices which are as accurate and precise as the ones computed with samples of $N = 1,000$ spectra. The underlying idea being that we can start with an input covariance matrix constructed from a small number (n) of high-resolution N-body simulations, and denoise that covariance using a ML method trained using a high (N) number of simplified mocks.

We simulated a grand total of $N_{max} = 30,000$ halo maps using ExSHalos, and we downloaded a total of $N_{max} = 15,000$ Quijote simulations, which then became (after matching with ExSHalos) our sample of $N_{max} = 15,000$ Sancho Panza halo catalogs. The spectra from these very large samples were used to compute the best case scenario, corresponding to the ideal cosmological covariance matrix for each simulation. It is important to say that we do not necessarily use this total number of spectra to train the ML suites: in each training we used 120 input matrices and 120 target matrices, however only the input matrices are completely independent. In other words, some targets matrices may be correlated with other targets, as a single target matrix can include many different input matrices. E.g., in the cases of $n = \{50, 100, 200\}$ input spectra we used $\{6,000; 12,000; 24,000\}$ spectra (of the "cheap" simulation, ExSHalos) for the entire training process.

Since the Quijote/Sancho Panza simulations were already available, the main limitation of our model was the computational cost associated with running the ExSHalos simulations, but in a realistic application of our method the cost of producing the mocks would be negligible compared with the cost of running the N-body simulations–and our denoiser is a tool for beating down that second, much more onerous cost.

4.4 The ML suite

The ML suite we used in this work was an image denoising algorithm, as as seen in Sections 3.3.2 and 3.3.3. Different ML models were constructed in order to deal with each combination of input and target matrices with (n, N) spectra. The size of the data set was composed with 120 matrices (240 in total, because each input matrix had its respective target). For each model we monitored the loss function for the MSE using 40 epochs each. Considering the size of the sets we have used 86 matrices in the train stage, 10 matrices for validation, and 24 matrices for test.

The CNN used as image denoising has its first two 2D convolutional layers, with

64 and 32 filters, corresponding to the encoder part. The decoder was composed by two transposed convolutional layers (respectively with 32 and 64 filters and their respective dropout layers). Lastly, these layers were followed by one convolutional layer with only one filter. Each convolutional layer, except for the last one, was followed by a dropout layer, for which we have chosen a rate of 0.05. The activation function for each internal layer was the LEAKY RELU ($\alpha = 0.001$), and for the last layers we used the hyperbolic tangent activation function. Finally, the batch size was 15, all the kernels have the size of (3×3) pixels, we have used the GLOROTUNIFORM initializer for the weights in each layer and, since we work with real-valued matrices, the input data has a single channel. The entire method was implemented with the help of the KERAS library (200).

We did not use *pooling* or UPSAMPLING layers in our network, since we verified that this operation tends to discard useful image details, as was also found by Reference (213). On the other hand, the use of dropout layers has indeed improved our results. Still in accordance with the findings of the aforementioned reference, the use of a small number of layers improved the model performance. Our choices of hyperparameters were made according to the best final performance in terms of the lowest MSE values as loss function for the training and test sets. We have tested other metrics as loss function (e.g. logarithm hyperbolic cosine, mean absolute error, and mean squared logarithm error), but the best results were obtained using the MSE. We also tested the denoising technique using a REDNet (213), as commented in Section 3.3.3, but the results were significantly worse compared with our standard architecture.

Covariance matrices often reflect a hierarchy of observables with different signals and different noises. In order to homogenize the entries of the covariance matrices in the training stage, we normalized the rows and columns of all the matrices using the diagonal of the matrix with the highest number of spectra available (i.e., $N_{max} = 30,000$ EXSHALOS spectra) according to:

$$Cov_{ij}^{(N)} \;\rightarrow\; \frac{Cov_{ij}^{(N)}}{\sqrt{Cov_{ii}^{(N_{max})}\, Cov_{jj}^{(N_{max})}}}. \tag{4.1}$$

For the EXSHALOS catalogs, the normalized $Cov_{ij}^{(N_{max})}$ becomes in fact the correlation matrix for that sample. This was discussed in Section 2.3.5.1, see Equation 2.35. Also, this normalization trick helped achieve a faster convergence of the ML method during the training stages[3]. We train our ML denoised on these normalized matrices, and plug back the normalization to recover the denoised covariance matrices. At the end of the whole process we have imposed the symmetry of the covariance matrices, $Cov_{ij} \rightarrow (Cov_{ij} + Cov_{ji})/2$.

[3] We have tested different normalization methods, for instance, utilizing directly the correlation matrices, or normalizations from the means of the diagonal of different matrices. However, the use of a fixed normalization for all the matrices proved to be the best strategy for the denoising method. We have also checked that it makes very little difference which precise normalization scheme is used: the results in terms of the parameter estimation remain basically the same.

Besides, for the different ML models for each combination (n, N) of input and target covariance matrices, we used four different random seeds for each model, accounting for the *epistemic error*. Therefore, we can account of at least part for statistical variations in our results, which are due to different solutions (different CNN final weights) found during the training process. In our tests we selected the best-performing of those solutions to apply on the input matrices from the Sancho Panza data set.

4.5 Results

In this section, we highlight the primary outcomes of our research by contrasting the input and denoised covariance matrices derived from the ExSHalos and Sancho Panza data sets. Initially, we assess the denoiser's efficacy in eliminating noise from the input covariance matrices. Here, we exclusively employ the ExSHalos data set for training, validation, and testing purposes. Subsequently, we move on to the pivotal step, wherein we apply the denoiser, trained using the ExSHalos data set, to the input covariance matrices of the Sancho Panza data set. We then compare the denoised matrices with the best-case scenario, which entails a covariance matrix derived from a sample comprising thousands of Sancho Panza simulations. This phase allows us to demonstrate the generalizing power of our approach, thereby illustrating its *robustness* (refer to Section 3.1), and serves as a proof-of-concept for the method.

We commence, in Section 4.5.1, with a visual representation of the matrices in their normalized form. Next, in Section 4.5.2, we quantify the MSE between the cosmological covariance matrices obtained during the training phase, which involves using the test subset of the ExSHalos matrices and all the trained models. Additionally, we analyze the ranked eigenvalues and diagonal values of the matrices to assess the loss of coherence due to noise and its recovery post-denoising in Sections 4.5.3 and 4.5.4, respectively. Furthermore, in Section 4.5.5, we introduce an analytical approximation for the random process underlying the estimation of sample covariance matrices, expressed in terms of the *Wishart distribution*. This analytical tool enables us to quantify the enhancement of the covariance matrices resulting from the denoiser's application. Finally, towards the end of this section (Section 4.5.6), we present MCMC estimations of the cosmological parameters under various scenarios. Through this analysis, we showcase the extent to which the denoiser enhances the accuracy and precision of the covariance matrices in terms of their ultimate product–the parameter constraints.

4.5.1 Visualizing the matrices

In Figure 24 we can already note the power of the ML method from the visual representations of these matrices, by comparing the normalized matrices (according to Equation 4.1) against the target and the best-case scenario, where the covariance was computed with many thousands (N_{max}) of spectra. The first row represents, from left to right, the input, target, denoised, and best covariance matrices of ExSHalos, for the model with input and target

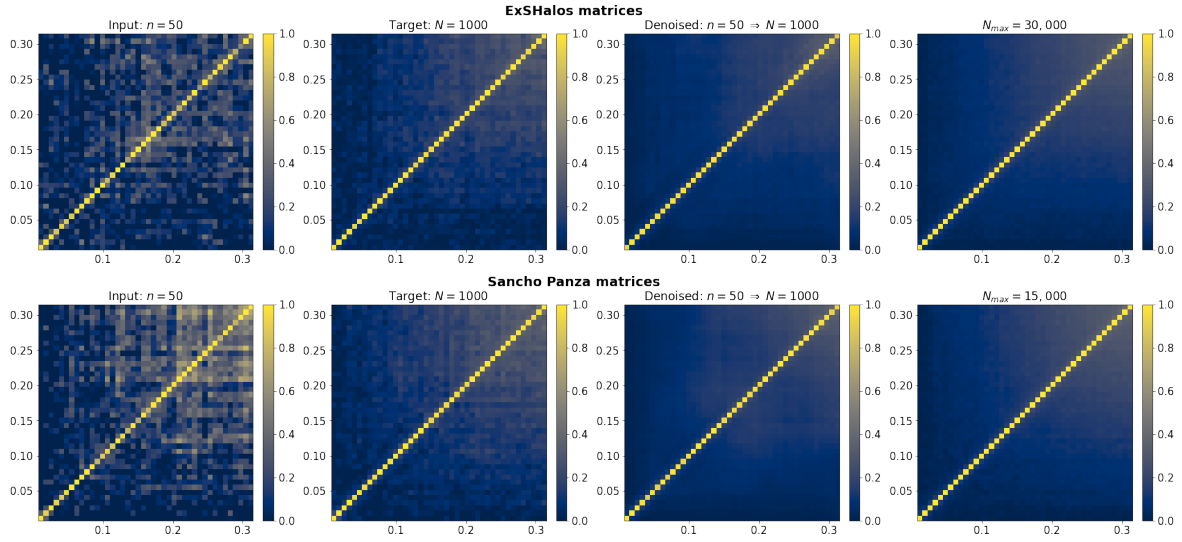

Figure 24 – **Comparison of the (normalized) cosmological covariance matrices**. We present a comparison for ExSHalos (first row) and Sancho Panza (last row). In the first column we have the input matrices with $n = 50$ spectra; the second column shows the target matrices, with $N = 1,000$ spectra; the third column contains the respective denoised matrix, corresponding to the model for the combination of input and target samples $(n = 50, N = 1,000)$; and the fourth column contains the *best* matrices, built using all the available spectra N_{max}. In each one of these figures, the axes corresponds to values of k, representing the 42 Fourier bins of the power spectrum. *Source:* Reference (40).

matrices of sample sizes $(n = 50, N = 1,000)$, respectively. The second row represents the same, but for the Sancho Panza data set. We stress, once again, that the denoiser is exactly the same as the one in the first row, since it was trained only with the ExSHalos data set.

It can be seen from the panels that the denoised matrices appear almost identical as the best ones, which were obtained with all the spectra of the entire data set ($N_{max} = 30,000$ for ExSHalos and $N_{max} = 15,000$ for Sancho Panza). This last feature can be explained by the choice of activation functions in the different layers, and because the ML models were trained using a huge number of different spectra, which appears to retain this information in the weights of the network. Therefore, the denoised matrices are visually smooth and noiseless, especially when compared with the original, $n = 50$ covariance matrices. At least from a purely visual standpoint, the power of the algorithm resides in the fact that the method is able to learn how to remove the noise using only ExSHalos matrices, and to apply this learning to the Sancho Panza ones.

A more accurate comparison of the matrices can be glimpsed from comparing slices (rows/columns) of the normalized covariance matrices, in order to show both the diagonal and off-diagonal elements (149). In Figure 25 we show a few fixed k_i slices of these matrices as a function of k_j, with the corresponding values for the input, denoised, target and *best* normalized covariances. Roughly speaking, all the matrices follow the behavior of the best-case scenario

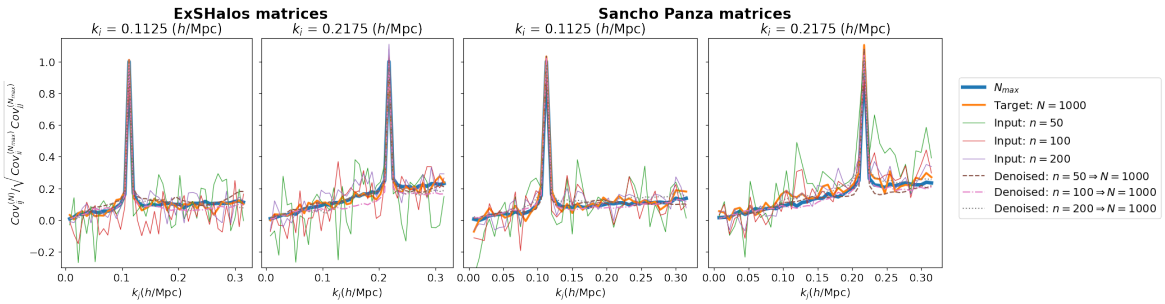

Figure 25 – **Comparison of slices of the normalized covariances.** This comparison was done according to Equation 4.1. We present ExSHalos (left panels) and Sancho Panza (right panels), for different fixed values of k_i. The peaks correspond to points along the diagonal. The plots show that the denoiser is able to remove the noise (seen in the input matrices), without introducing new features, so the denoised matrices match closely the targets and the *best* matrix (with a sample of N_{max} spectra). *Source:* Reference (40).

(N_{max}), but it is clear that the input matrices are severely affected with noise, especially in the off-diagonal elements and particularly for $n = 50$ and $n = 100$. However, after applying the ML denoiser those fluctuations basically disappear, and the off-diagonal elements match the behavior of the target and *best* matrices in all cases, even for $n = 50$. Moreover, from these plots it can also be seen that the off-diagonal structures in the ExSHalos matrices scale differently with k compared with those of the Sancho Panza matrices, especially on small scales. However, the denoiser (which was trained only with the ExSHalos matrices) is able to properly recover the off-diagonal behavior of the Sancho Panza matrices as well, which is further evidence for the generalization power of the ML method.

4.5.2 The MSE between different matrices

Although the visual inspection of the previous subsection hints at the good performance of the method, we have monitored improvements in the covariance matrices using the MSE metric:

$$\mathrm{MSE} = \frac{1}{\mathcal{N}} \sum_{l=1}^{\mathcal{N}} \frac{1}{n_k^2} \sum_{i,j=1}^{n_k} (Cov_{ij}^{l} - Cov_{ij}^{(N_{max})})^2 \ , \tag{4.2}$$

where Cov_{ij}^{l} are the input, target, or denoised covariance matrices, $Cov_{ij}^{(N_{max})}$ is the best matrix (produced with the entire data set), n_k the number of bins of k in the data vector, and $\mathcal{N}$ is the total number of matrices used in the evaluation. We computed the MSE only for the test subset of the ExSHalos matrices, and the results are shown in Figure 26. The gray line corresponds to the MSE for the original (input) matrices, the blue line corresponds to the denoised matrices, and the error bars account for the standard deviation for the results for each random seed. As a comparison, the black line represents the MSE of the target matrices ($N = 1,000$), which is a lower bound for the MSE of the original matrices and provides a useful sanity check. The

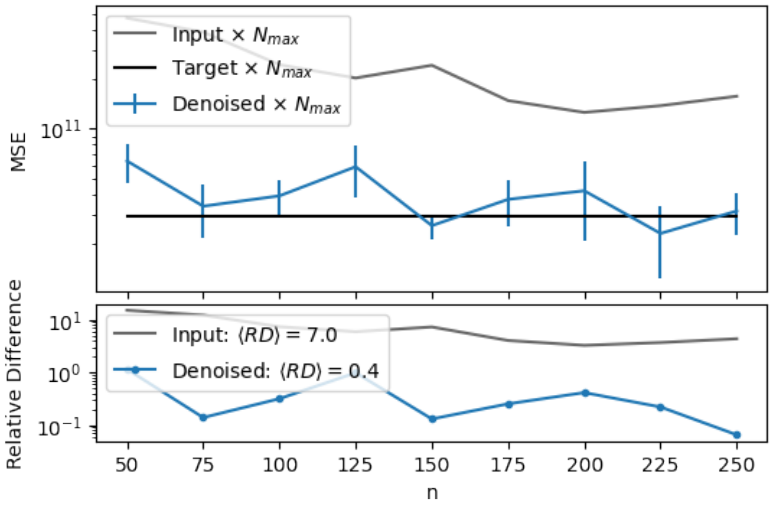

Figure 26 – **MSE comparison for the cosmological covariance matrices.** This comparison was done computing the best sample ($N_{max} = 30,000$) from ExSHALOS and the original (input) matrices, in gray, and the denoised ones, in blue. As the sample size of the input matrices (n) grows, the agreement between the matrices improve and the MSE becomes smaller. The MSE between the best matrix and the target matrix (with $N = 1,000$ spectra) is shown as the black line. The error bars account for the values obtained for different seeds of the same model. The lower panel shows the relative difference according to Equation 4.2. *Source:* Reference (40).

lower panel in Figure 26 shows the residue:

$$\text{Relative Difference} = \frac{abs\left(\text{MSE}_{\text{Y}\times N_{max}} - \text{MSE}_{\text{Target}\times N_{max}}\right)}{\text{MSE}_{\text{Target}\times N_{max}}}, \tag{4.3}$$

where Y stands for input or denoised.

Naturally, the MSE decreases when the sample size (n) grows. The decrease in MSE that results from applying the denoiser is much larger: the mean values of the residue ($\langle RD \rangle$) show that the denoised matrices deviate by only ~ 0.4 from the best matrices, compared with approximately 7.0 for the original input matrices–a factor of more than 17 improvement. Moreover, the MSE of the denoised matrices are weakly dependent on the original sample size, which may be indicative that even with an original sample of only $n = 50$ spectra the denoiser is already able to eliminate most of the noise of the covariance matrices, and increasing the sample size does not significantly improve upon that noise reduction. Note that the black line (MSE of the target matrix) is reached in some cases by the denoised matrices, which is not completely unexpected since the training of the ML method includes more matrices than are included in the targets.

4.5.3 The eigenvalues of the matrices

A complementary analysis to the one above can be made to compare the accuracy with which we reproduce the cosmological covariance matrices, which relies on the eigenvalues

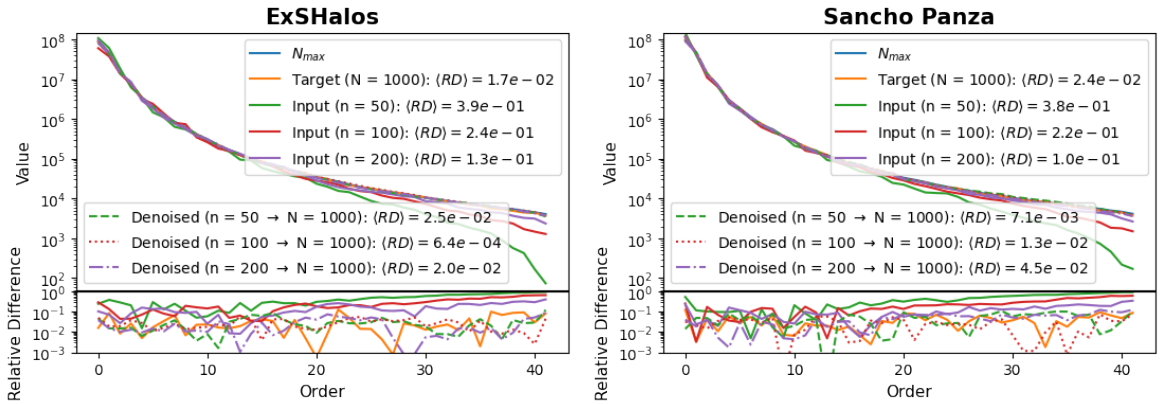

Figure 27 – **Comparison of the ranked eigenvalues and their relative difference.** The ranked eigenvalues are presented in the main plot, while their relative difference (with the respective values for the matrix with N_{max}), in the subplot. For the ExSHalos, the results are presented on the left, and for Sancho Panza on the right. *Source:* Reference (40).

of those matrices. Typically, the eigenvalues that effectively carry information obey some power law, and as we reach the lower eigenvalues the noise appears as an abrupt change in the scaling of the eigenvalues (268, 269). In Figure 27 we show the ranked eigenvalues for covariance matrices from ExSHalos (left panel) and Sancho Panza (right panel). We can see that the eigenvalues of the denoised matrices are much closer to the target and best matrices, while the original input matrices show clear signs of noise in the lower end of the spectrum of eigenvalues, which become more important as we decrease the sample size. We quantify the difference in eigenvalues with respect to the best covariance matrix in terms of the same relative difference that was defined in Equation 4.3. For ExSHalos, the improvements are a factor of more than ~ 10, while for Sancho Panza the improvements are between ~ 50 (for $n = 50$) and more than ~ 10 (for $n = 100$).

4.5.4 The diagonal values of the matrices

As an additional check to ensure that the denoised matrices match the target or the *best* matrices, we have also looked at the values of the diagonals of those matrices. In Figure 28 we show that comparison for the *best* matrices (computed with a sample of N_{max} spectra), the target ($N = 1000$), as well as the input and denoised matrices in the cases of samples $n = [50, 100, 200]$. In both cases (ExSHalos and Sancho Panza) the diagonals of the denoised matrices are a much better match to the diagonal of the *best* covariance matrix, with relative differences lower than 0.1. This is in contrast with the input (noisy) matrices, which even for $n = 200$ still show deviations greater than 10% compared with the *best* matrix. This result, combined with the comparison of the eigenvalues of Section 4.5.3, shows that the denoiser is able to recover the key features of the matrices, leading to denoised covariances which are at least as good as the target, and often come very close to the best case scenario.

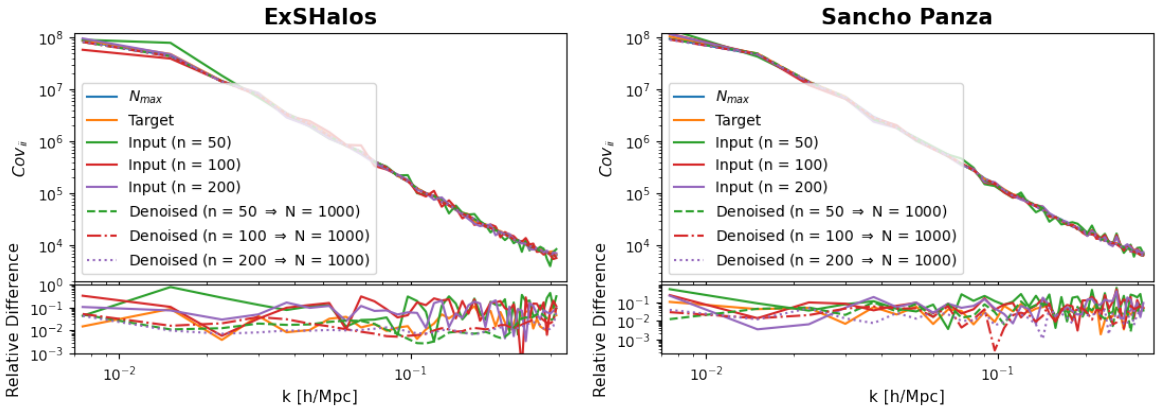

Figure 28 – **Comparison of the diagonal values of the covariance matrices.** The comparison is done for the *best* case scenario (N_{max} spectra), target ($N = 1,000$), input, and denoised with $n = [50, 100, 200]$. The relative differences (lower subplot) are computed with respect to the *best* matrices. *Source:* Reference (40).

4.5.5 An analytical comparison for the machine learning black box

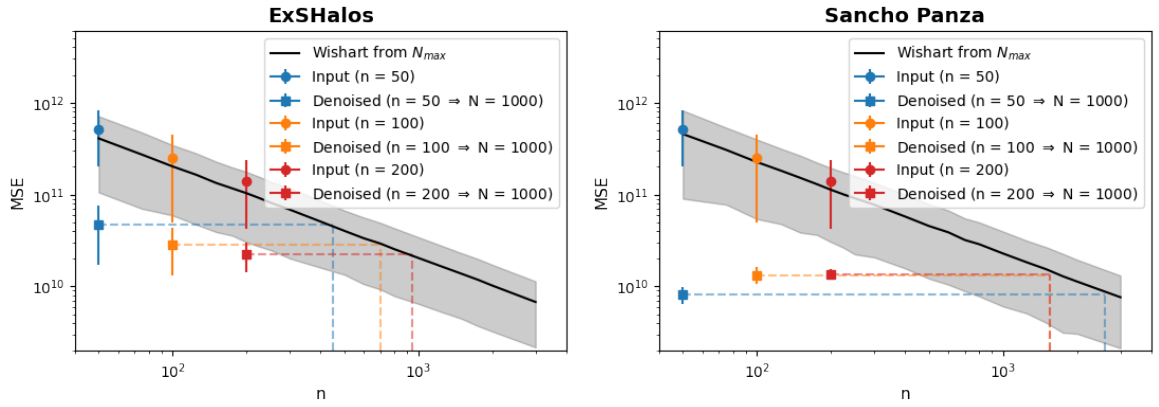

Figure 29 – **Wishart analysis.** MSE comparison between the best ExSHalos (on the left panel) and Sancho Panza (on the right panel) matrices with: matrices estimated from the Wishart distribution, in black; the input, in circles (for $n \in [50, 100, 200]$); and denoised matrices, in squares (from $n \in [50, 100, 200] \Rightarrow N = 1000$). The colors corresponds to the number of matrices in the input and resulted denoised matrices. The gray region corresponds to 1σ deviation for the mean values, in the case of the Wishart matrices. The dashed lines have the intention to guide the reader to see to which number of spectra n the input matrices were taken to their Wishart comparison. *Source:* Reference (40).

The encouraging results above indicate that the denoiser is effectively learning about the specific patterns of signal and noise, producing covariance matrices that are equivalent to those computed with samples of a much greater size than the sample size of the covariance that we plug into the denoiser. In other words, applying the denoiser seems equivalent to increasing the sample size.

In order to check this conjecture we can compare the resulting (denoised) matrices with a model for the probability distribution function behind covariance matrices, and which

describes how their fluctuations depend on the sizes of the data vector and the sample size. There is, in fact, such an analytical description for covariance matrices in terms of the *Wishart distribution* (37, 270):

$$p(\hat{M}|M,\nu,\eta) = \left(\frac{\nu^{\nu\eta/2}|M|^{-\nu/2}|\hat{M}|^{\gamma/2}}{2^{\nu\eta/2}\Gamma_\eta[\nu/2]}\right)\exp\left[-\frac{\nu\mathrm{Tr}\left(\hat{M}M^{-1}\right)}{2}\right] . \tag{4.4}$$

Here, M represents the statistical mean of the matrices, $|M|$ is its determinant, and $\hat{M}$ is the random variable–in this case, the sample covariance matrix. The parameters of this distribution are the size of the data vector η (the dimension of the matrices is $\eta \times \eta$) and the number of degrees of freedom ν (in our case, the sample size). In the formula above, $\gamma = \nu - \eta - 1$ and $\Gamma_\eta[\nu/2]$ is the multivariate Gamma function. Therefore, given an "ideal" covariance matrix M, this distribution allows us to generate random covariance matrices corresponding to different sample sizes. We should also point out that the Wishart distribution is unbiased, since $\langle\hat{M}\rangle = M$.

The Wishart distribution is useful in this context since it provides a model comparison for the statistical fluctuations of covariance matrices as a function of sample size ($\nu \to n$). This means that we can infer what is the effective sample size of the denoised matrices by analyzing their MSEs, and comparing that with the result from the Wishart distribution where the "ideal" covariance matrix is computed using the maximum number of spectra available (N_{max}). In Figure 29 we plot the MSE (mean and variance) between the input and covariance matrices, compared with the covariance matrix obtained with all the spectra available ($N_{max} = 30,000$ for ExSHalos and $N_{max} = 15,000$ for Sancho Panza). The Wishart distribution (mean and variance) is denoted by the black line and gray region. The input matrices are represented by the circles and their respective 1 σ deviations, while the denoised matrices are represented by squares, both in the cases $n \in [50, 100, 200]$. Since the mean MSE values for the denoised matrices are significantly lower compared with the input matrices, according to the Wishart distribution the denoiser is effectively taking matrices with samples of $n = 50 - 200$ spectra, and transforming them into covariance matrices with much larger samples.

For the matrices from ExSHalos, the denoised matrices with $n \in [50, 100, 200]$ are closer to the prediction from the Wishart distribution for $n \in [450, 700, 940]$ spectra, respectively. In the case of the Sancho Panza matrices, that effective sample size is even higher: $\sim 1550 - 2500$. The denoised matrices also have a lower scatter, especially in the case of Sancho Panza, which means that the chance that a denoised sample covariance ends up producing a poorly estimated covariance matrix is lower than that for the equivalent effective sample size without denoising.

It is also interesting to note that the Sancho Panza matrices benefit more from the denoiser. This may be due to the fact that the matrices from Sancho Panza are already smoother than the ExSHalos matrices, resulting in better estimations to begin with. We have made

tests changing the number of spectra in the normalization matrix of ExSHalos (to train the models), but still, the Sancho Panza results were better than for ExSHalos. Therefore, the normalization is not the cause behind this effect.

The interpretation presented in this section relies on a simple treatment of the covariance matrix, which is regarded as a quadratic combination of Gaussian random variables (in our case, the spectra). A more thorough analysis can be made in terms of the likelihood, which takes into account a marginalization over the inverse Wishart distribution, leading to a modified multivariate $t-$distribution instead of a Gaussian distribution for the data (271). Similarly, using the Wishart distribution to propagate the uncertainty in the theoretical model, a Bayesian approach can be used to combine simulated and theoretical covariance matrices, also reducing the number of simulations required to reach some threshold of precision and accuracy (272).

4.5.6 Recovering the cosmological parameters

Finally, it is important to analyze the ability of the denoised matrices to recover the fiducial simulated parameters and compare these estimation with the parameters coming from the original matrices to validate our results. The analysis is presented in Figure 30. We have explored the parameter space using the MCMC approach, implemented with the help of the `emcee` library (273), utilizing as data points some random power spectrum vector, and using in each case the different cosmological covariance matrices. Our goal is to study the multivariate probability distribution function for the parameters: H_0, Ω_b, Ω_c and the nuisance parameter bias b, for the matrices built with the maximum number of spectra, target, denoised, and input matrices. In all the analyses we have used 20 walkers and chains of $5,000$ length (except for the input matrices, for which we have used $6,000$).

Overall, in all models (for $n = 50$ or $n = 100$, and $N = 1,000$) and both sets of matrices (ExSHalos and Sancho Panza) all the parameters were well constrained using the denoised matrices. It is interesting to see that the inference considering the input matrices have a "false" precision (the volumes in the parameter space are lower, when compared to all the other matrices) and is very inaccurate, because the mean values estimated is sightly (for $n = 100$) and highly (for $n = 50$, specially in the ExSHalos input matrix) shifted. Moreover the input matrix with $n = 50$ and $n = 100$ spectra presents fluctuations on their contours, which are improved/removed in the denoised matrix estimation.

Quantitatively, the improvements in the mean values becomes clear when we compare the bias in the expectation value of a parameter in units of its variance:

$$\mathbf{\Delta} = \frac{\Delta\mu}{\sigma_{N_{max}}} = \frac{\text{abs}\left(\mu_X - \mu_{N_{max}}\right)}{\sigma_{N_{max}}}, \tag{4.5}$$

where μ corresponds to the mean of a parameter, X represents the input and denoised matrices, and $\sigma_{N_{max}}$ is the standard deviation of the parameter obtained using the *best* covariance matrix.

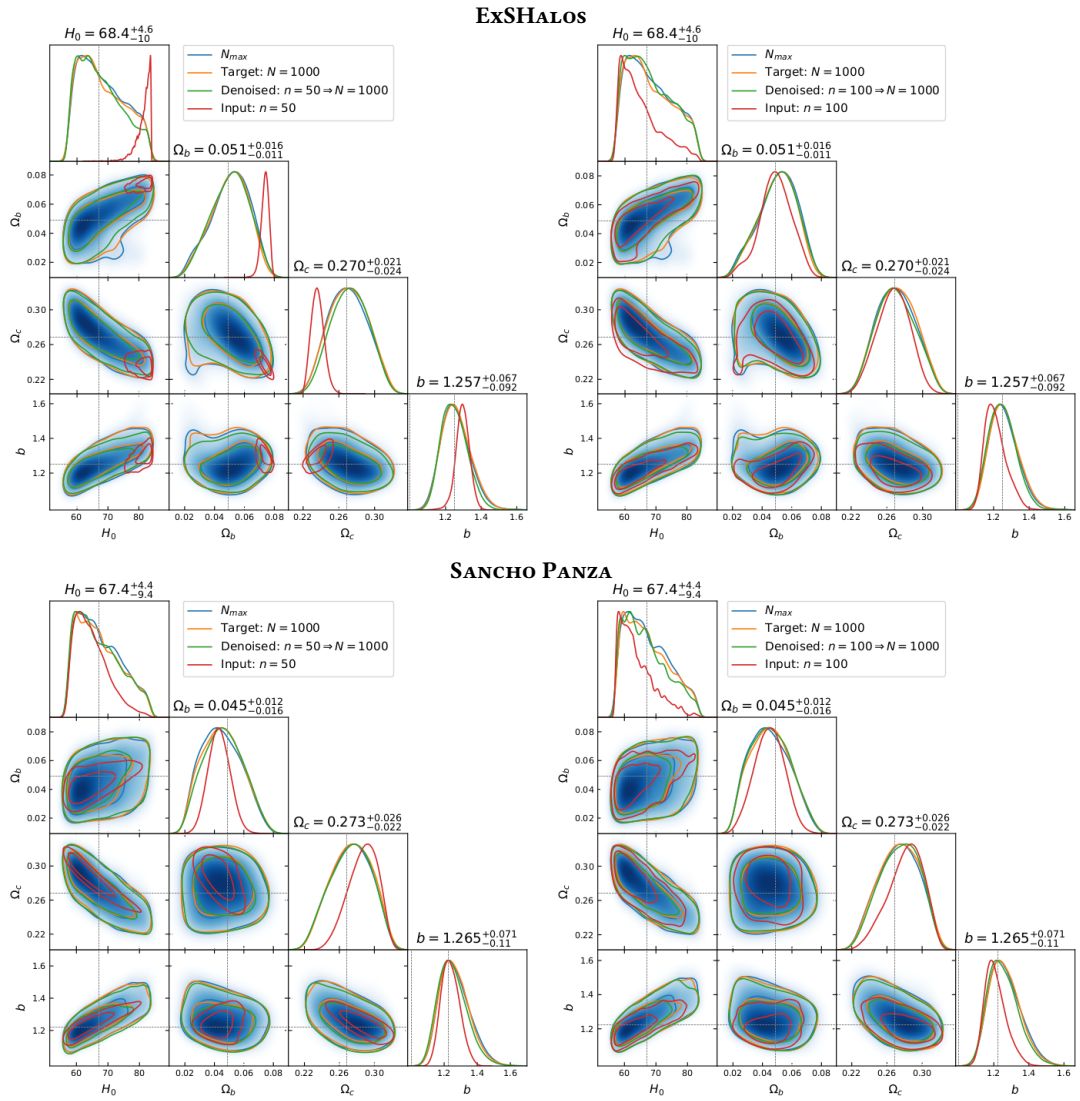

Figure 30 – **Cosmological parameter estimation comparison**. Comparison for different covariance matrices from ExSHalos (upper panels) and Sancho Panza (lower panels): matrices built using all the spectra available (with $N_{max} = 30,000$ spectra for ExSHalos, and $N_{max} = 15,000$ spectra for Sancho Panza); targets ($N = 1,000$ spectra); input (using $n = [50, 100]$ spectra); and the denoised ones ($n \Rightarrow N$). *Source:* Reference (40).

The results for ExSHalos and Sancho Panza matrices are presented in Table 2, in the scenarios where the input and denoised matrices are based on samples of $n = [50, 100, 200]$ spectra.

In the case of ExSHalos, the results of this quantity from the bad to the denoised matrices represent the improvements that is ~ 128 times, in the case of Ω_b, for models with $n = 50$; ~ 13.6 times, in the case of Ω_c, for the models with $n = 100$; and ~ 17.9 times, in the case of H_0, for models with $n = 200$. For matrices from Sancho Panza, the same comparison follows to ~ 68 times, for Ω_c, for models with $n = 50$; ~ 5.4 times, in the case of Ω_c, for $n = 100$; and ~ 13.1 times, in the case of Ω_b, for $n = 200$. The only parameters which are not improved are: Ω_b, in the case of the model using $n = [50, 100]$; and b, for the model using $n = 200$, in the Sancho Panza analysis. Notwithstanding, it is clear the improvement power

Table 2 – **Bias in the estimated parameters.** It is presented in units of the variance ($\mathbf{\Delta}$), measured with respect to the parameters determined using the covariance matrix built using all the spectra available (N_{max}).

	Parameter	ExSHalos		Sancho Panza	
		Input	**Denoised**	**Input**	**Denoised**
n = 50	H_0	1.810	0.102	0.380	0.041
	Ω_b	1.754	0.014	0.107	0.112
	Ω_c	1.575	0.104	0.615	0.009
	b	0.470	0.089	0.293	0.144
n = 100	H_0	0.463	0.044	0.500	0.095
	Ω_b	0.118	0.021	0.002	0.043
	Ω_c	0.132	0.010	0.357	0.066
	b	0.552	0.092	0.605	0.125
n = 200	H_0	0.135	0.008	0.063	0.012
	Ω_b	0.151	0.054	0.125	0.010
	Ω_c	0.163	0.063	0.096	0.019
	b	0.424	0.032	0.027	0.035

of the proposed suite: using only about a hundred spectra for an input matrix, the denoiser can achieve results as if this matrix was made with thousands of spectra. Moreover, the results for Sancho Panza completes the proof of the power of generalization of the proposed suite.

4.6 Discussion and Conclusions

The search for more efficient ways to compute the highly accurate cosmological covariance matrices that are now needed in Cosmology has many different approaches (36, 37, 139–146, 250, 252, 254–258). The main goal of this field is to provide precise matrices for parameter inference, that can be used in MCMC explorations of the likelihood, and result in unbiased estimates of the probability distribution function of the parameters. The results of these analyses can lead to improved cosmological probes and/or experiments, for different parameters; they can be used to break the tension of some parameters (as in the case of Hubble parameter (14)); as well as in the search for the nature of DE.

This work presents an efficient approach for the estimation of cosmological covariance matrices. The main idea behind our method is that, starting from matrices built with only hundreds of spectra, we are able to provide covariance matrices that are as good as if they were built with thousands of spectra. We have implemented this method using CNNs as a denoising algorithm, that cleans the noise in the input matrices. Visual inspection (see Figures 24 and 25) already shows that the noise was removed, without the introduction of any visible artifacts when compared with the best matrices.

The ML was trained in a data set of cosmological covariance matrices built with a mock generator of halo catalogs, called ExSHalos (146). The reason behind this choice is the

fact that this code was already designed to be extremely fast, when compared to other mock generators or N-body simulations, and the ease of applying halo-finders in that context. We were able to quickly generate many thousands of different maps, allowing us to build a very large data set of power spectra with which we could build our covariance matrices. In practice, the main limitation of our model comes from the computational cost associated with running the ExSHalos simulations. However, we should also note that the number of catalogs that were needed is rather small, since very good results can be obtained already for the model with $n = 50 \Rightarrow N = 1000$, which needs only $6,000$ ExSHalos maps. Moreover, as long as the ML models are trained, highly accurate covariance matrices from samples of N-body simulations can be obtained in a matter of seconds.

Although the ML suite was built using matrices coming from an approximated method, the main objective of this work was to test the generalization power of the suite when applying the models to matrices coming from N-body simulations such as Quijote (112). Hence, we are interested in taking the cleaning ML process learned with the covariance matrices from the approximated method, and applying it to matrices coming from the best simulations available. In this way, we expected to provide matrices that can even circumvent the well known errors (up to 10%) in the parameter inference, that occur when using matrices that were derived only from mocks or other approximated methods (149).

In order to check whether the resulting (denoised) covariance matrices were more similar to the best ones, we performed a series of tests. We started by computing the MSE (see Equation (4.2)) and verified that the denoiser is able to reduce that indicator by a factor of ~ 10–see Figure 26. We also compared the ranked eigenvalues and looked at the diagonal values of the matrices, and showed that after denoising the input matrices we recover the main features of the *target* and even of the *best* matrices, which are built using the maximum number of spectra available (N_{max}).

An interesting point of debate is whether the predictions of a ML model can be compared with a simple mathematical model. We have shown that the proposed method can be matched in terms of an extrapolation according to the Wishart distribution (37, 270), by effectively augmenting the size of the sample that underlies the covariance matrix. Namely, in the case of the ExSHalos data set, the denoised covariance matrices built from an initial sample of $n = 50 - 200$ spectra are equivalent to the ones computed using $\sim [450 - 940]$ spectra. In the case of Sancho Panza the denoised matrices are similar to the ones computed with $\sim [1550, 2500]$ spectra. The former result was expected, since the effect of the denoiser was to bring the matrices very close to the target matrices (with samples of $N = 1,000$). The latter result can be explained by the fact that the *best* matrix built with $N_{max} = 15,000$ Sancho Panza spectra has half as many spectra when compared ExSHalos (for which $N_{max} = 30,000$), and because those matrices are, indeed, smoother than the ones from ExSHalos.

Ultimately, the strongest evidence for the power of the denoising technique is provided

by the parameter estimation, in particular the plots in Figure 30, and the analysis of the bias as expressed by $\mathbf{\Delta}$–see Equation 4.5 and Table 2. All the parameters (H_0, Ω_b, Ω_c and b) were well constrained in the case of the denoised matrices. We see improvements (when comparing the input and denoised matrices) for all parameters in the case of ExSHalos, and for most parameters in the case of Sancho Panza. In particular, the improvements achieved for H_0 using the sample with $n = 50$ spectra were of a factor of ~ 17.9 in the case of ExSHalos, and of ~ 9.4 in the case of Sancho Panza. Only when a parameter was already very well determined using the input covariance matrices ($\Delta \lesssim 0.1$), the denoised matrices did not lead to further improvements. These results form the basis for establishing the predictive power of the proposed suite, as well as the generalization of the methods, which can be trained in covariance matrices from mocks and applied to covariance matrices from N-body simulations.

To summarize, in this work we have demonstrated the power of image denoising techniques as a new approach to improve sample cosmological covariance matrices. Our work included some modeling of real-life effects that are analogous to observational conditions (in terms of a non-trivial mask). Besides the architecture presented here, we have also tested different network configurations, different methods to normalize the matrices, and even the residuals from the encoder-decoder networks (213). In our tests, the architecture presented was the one which provided the best results.

The next steps regarding this project are mainly related to the future prospects for applying our methodology to real data. However, before doing so, some checks should be performed: (i) test the machinery in matrices from different cosmologies besides the fiducial one (which was used in the training stage), in order to check if the method can be generalized in that sense; (ii) investigate whether other ML image denoising methods can result in even better covariance matrices; and (iii) apply the ML suite to more complex, larger and more realistic covariance matrices.

5 MIMICKING THE HALO-GALAXY CONNECTION

Understanding the intricate relationship between the baryonic and DM components of the Universe has increasingly become a focal point in Cosmology and galaxy evolution research. Currently, the key questions revolve around characterizing the connections between galaxy properties and the DM halos inside which they form and evolve, within the cosmological framework of formation of the large-scale structures (172).

As briefly discussed in Section 2.5.2.1, various methods are employed to investigate and characterize this connection. These include SHAMs (176), decorated HODs (177), SAMs (173–175), and state-of-the-art *hydrodynamical simulations* (154–158). Despite the diversity of these approaches, a method capable of quickly providing the detailed insights of hydrodynamical simulations at the data handling level is still lacking.

In this context, ML techniques are emerging as a promising avenue (53, 53–56, 58, 60, 61, 64, 184, 190, 274, 275), which could be considered a halo-galaxy connection technique in itself, given that the outcome of the analysis is a set of models characterizing those links. Predicting stellar mass, which has a strong correlation with halo mass, is relatively straightforward. However, other properties such as color or star formation rate (SFR), which are influenced by numerous factors related to secondary halo properties (other than the mass itself), often in non-trivial ways, remain challenging to reproduce even with sophisticated ML methods.

Many applications have employed the Illustris simulations as a data set, which has only heightened the relevance of those challenges. For example, in Reference (56), authors utilized ERTs to predict various simulated galaxy properties, including gas mass, stellar mass, BH mass, SFR, and color. More recently, several studies have focused on predicting stellar mass alone using CNNs (54, 55, 58). These analyses have extended to measuring clustering properties and comparing them with halo-galaxy connection techniques such as HODs. Additionally, in Reference (53), ERTs were employed to predict both stellar mass and SFR. The trained machine was then applied to the MultiDark DM-only simulation, and the results were compared with a SAM to validate the extrapolation power of the model. However, these pioneering works failed to reproduce accurately the diversity of the galaxy populations, or the way in which they are related to their hosting halos.

In this chapter, we utilize the largest IllustrisTNG simulation box, TNG300 (with a side length of $205\,h^{-1}$ Mpc), as our data set. The halo-galaxy connection is established by predicting properties of central galaxies, including stellar mass, half-mass radius, specific star formation rate (sSFR), and color, based on five halo properties: halo mass, concentration, spin, age, and a proxy for the overdensity halo environment. We adopt two different approaches:

- **Stacking raw and augmented models.** We assess the performance of various ML meth-

ods, including ERTs, κNNs, LGBM, and NNs (as discussed in Sections 3.2.2.1, 3.2.1, 3.2.2.2, and 3.3.1, respectively). Instead of relying on a single method, we combine predictions from different methods to generate a new data set. This combined data set is then used to train another ML model, producing a new, integrated prediction for each galaxy property. This "stacked" approach considers both the strengths and weaknesses of individual models (see Section 5.3 for further details). Additionally, due to suboptimal predictions for the tail of the distributions, we incorporate the SMOGN data augmentation technique (also discussed in Section 3.5.1) into our pipeline. The main achievements related to this work are associated with the publications of References (65, 276).

- **Converting regression to classification.** Recognizing the complex, interrelated processes involved in galaxy formation and evolution, we acknowledge that galaxy properties cannot be accurately determined solely by halo properties. To address this challenge, we introduce a model that accounts for uncertainties arising from the stochastic aspects of galaxy formation. Our approach aims to return joint probability distributions for the galaxy properties, rather than a single estimate for their values, given a set of halo properties. This is achieved using NNs by converting the original regression problem into a classification problem (as outlined in Section 3.4.2). Additionally, this work is related to the publication in Reference (69).

We present both these efforts in the Subsections 5.3 and 5.4, respectively.

While we propose to address the issue of stochasticity in the context of the halo-galaxy connection primarily through various applications of ML, other authors advocate for the use of alternative physical aspects such as *merger trees* (66, 277). However, neither of these methods represents the definitive solution to the problem, necessitating a thorough comparison between them (as discussed in Section 5.4.2.3). Furthermore, the majority of these methods only estimate galaxy properties for central galaxies, lacking information about the clustering of satellites and their properties, which are even more prone to scatter compared to their host halo characteristics (172). Consequently, there remains much to be explored before asserting ML methods as a comprehensive solution within the area of halo-galaxy connection.

5.1 The IllustrisTNG data

This analysis is grounded in the IllustrisTNG magnetohydrodynamic cosmological simulation, extensively documented in References (278–284). Utilizing the Arepo moving-mesh code (159), the IllustrisTNG suite represents an improved version over its predecessor, the Illustris simulation (285–287). Notably, the updated IllustrisTNG subgrid models incorporate a number of mechanisms such as star formation, radioactive metal cooling, and chemical enrichment from various stellar sources including SNII, SNIa, and AGB stars. Additionally, they incorporate feedback from stellar and super-massive BH activity. These models have been

meticulously calibrated to match an array of observational constraints, encompassing the $z = 0$ galaxy stellar mass function, cosmic star formation rate (SFR) density, halo gas fraction, galaxy stellar size distributions, and the black hole–galaxy mass relation.

In our study, we gauge the accuracy of our modeling by examining large-scale structure galaxy clustering. Hence, we have opted to analyze data from the largest available box in the database, IllustrisTNG300-1, henceforth referred to as TNG300. Spanning a side length of $205\ h^{-1}$Mpc with PBCs, TNG300 tracks the dynamical evolution of 2500^3 DM particles, each with a mass of $4.0 \times 10^7\ h^{-1}\mathrm{M}_\odot$, and initially, 2500^3 gas cells, each with a mass of $7.6 \times 10^6\ h^{-1}\mathrm{M}_\odot$. Renowned for its capability to replicate numerous observational measurements, TNG300 serves as a valuable resource for galaxy formation and clustering research, as demonstrated in References (278, 284, 288–293).

DM halos within the IllustrisTNG simulation are identified using a FoF algorithm, employing a linking length parameter $b = 0.2$ times the mean inter-particle separation (264). Subsequently, the gravitationally bound substructures, which we refer to as subhalos, are identified using the SubFind algorithm (134, 135). Subhalos that contain a non-zero stellar mass component are categorized as galaxies.

5.1.1 Halo Properties

In this work, we consider the following halo properties from the TNG300 simulation box:

- **Halo Mass.** Virial mass M_{vir} $[h^{-1}\mathrm{M}_\odot]$, computed by summing the mass of all gas cells and particles enclosed within a sphere of radius R_{vir}. This sphere is defined such that the enclosed density equals 200 times the critical density.

- **Halo Age.** Described in terms of a formation redshift $z_{1/2}$, which is defined as the redshift at which half of the present-day halo mass has been accreted into a single subhalo for the first time. For this computation, we utilize the progenitors of the main branch of the subhalo merger tree computed with SubLink, initialized at $z = 6$.

- **Halo Spin.** Defined as in Reference (294):

$$\lambda_{\mathrm{halo}} = \frac{|J|}{\sqrt{2} M_{\mathrm{vir}} V_{\mathrm{vir}} R_{\mathrm{vir}}}, \tag{5.1}$$

  where J is the angular momentum of the halo and V_{vir} is its circular velocity at the virial radius R_{vir}.

- **Halo Concentration.** Defined in the standard way as:

$$c_{\mathrm{vir}} = \frac{R_{\mathrm{vir}}}{R_{\mathrm{s}}}, \tag{5.2}$$

where R_s is the scale radius derived from fitting the DM density profiles of individual halos with a NFW profile (295).

- **Halo Overdensity on a 3 h^{-1}Mpc scale.** Defined as the number density of subhalos δ_3 within a sphere of radius $R = 3h^{-1}$Mpc, normalized by the total number density of subhalos in the TNG300 box (288, 296).

5.1.2 Galaxy Properties

Galaxies (i.e., subhalos with non-zero stellar components in TNG300), are characterized using the following properties:

- **Stellar mass.** M_* [$h^{-1}\mathrm{M}_\odot$], is defined here as the total mass of all stellar particles bound to each subhalo.
- **Specific star formation rate (sSFR).** Star formation rate, SFR [$\mathrm{yr}^{-1}M_\odot$] is computed as the sum of the star formation rate of all gas cells contained in each subhalo. Then, sSFR [$\mathrm{yr}^{-1}h$], is defined simply as the SFR per unit stellar mass: sSFR = SFR/M_*.
- **Galaxy Radius.** Stellar (3D) half-mass radius $R^{(*)}_{1/2}$ [h^{-1} kpc] is defined as the comoving radius containing half of the stellar mass of each subhalo.
- **Galaxy (g-i) color.** Is derived from the magnitudes provided at the IllustrisTNG data base. These magnitudes are computed by summing up the luminosities of all stellar particles of each subhalo (see Reference (297)). The IllustrisTNG magnitudes are intrinsic, meaning the attenuation produced by dust is not included.

5.1.3 Data pre-selection

As mentioned previously, our analysis focuses exclusively on central galaxies from TNG300, simplifying the modeling of the halo-galaxy connection. To avoid biasing our results with unphysical values, we have applied several data cuts. Firstly, only halos with masses above $\log_{10}(M_{\mathrm{vir}}\ [h^{-1}\mathrm{M}_\odot]) = 10.5$ are considered. Secondly, a minimum stellar mass of $\log_{10}(M_*\ [h^{-1}\mathrm{M}_\odot]) = 8.75$ is imposed, ensuring that halos contain at least 500 DM particles and galaxies have at least 50 stellar mass particles. This selection process yields a final galaxy sample of $174,527$ objects. Among this sample, approximately 48% are designated for training, 12% for validation, and 40% for testing.

However, analyzing the SFR and sSFR poses challenges in TNG300, as approximately 14% of galaxies at $z = 0$ have an SFR of exactly zero. This condition does not necessarily indicate quiescent galaxies, typically defined with $\log_{10}(\mathrm{sSFR}[\mathrm{yr}^{-1}h]) \sim -10.5$ in TNG300. To address potential numerical issues caused by these galaxies with null SFR, we assign them an artificial SFR. This artificial value is randomly sampled from a Gaussian distribution with parameters $\mu = -13.5$ and $\sigma = 0.5$, as described in a similar approach in Reference (298).

While this approach ensures a well-defined sSFR for all galaxies, predicting the assigned values statistically remains challenging, as discussed in the following sections[1].

5.2 Performance Metrics

The metrics provide a mean of measuring the difference between ML predictions $\hat{y}$ and the target values y. In this work we made use of MSE (see Equation 3.6) and:

- **Pearson Correlation Coefficient (PCC):**

$$\mathrm{PCC} = \frac{\mathrm{cov}\left(\hat{y}, y\right)}{\sigma_{\hat{y}}\sigma_{y}} \tag{5.3}$$

  which measures the Pearson correlation between the predictions and true values, being as good as close to ± 1 and as bad as close to 0.

- **Kolmogorov-Smirnov test (K-S test):**

$$D = \max\left(\left|F_1(x_1) - F_2(x_2)\right|\right), \tag{5.4}$$

  where $F_i(x_1)$ and $F_i(x_2)$ are two cumulative distributions (41). In essence, the K-S test measures the maximum distance between them. It is also useful to compute the 2D K-S test. The method is essentially the same as for the 1D K-S test, but accounting for 2-dimensional data. This algorithm, developed mostly for astronomical analyses, compute the cumulative distributions along the coordinate axes of the two variables. More details can be found in References (299, 300). In this work we have used the Reference (301) repository. For both versions of this test, the values are as better as close to 0 they are.

5.3 Stacking raw and augmented models

The methodology of this approach, outlined in Figure 31 and also discussed in Reference (65), consists of several steps. Firstly, the input data is prepared as described in Section 5.1.3. Once the input data set is ready, two different paths are taken:

- **Path 1 (top of Figure 31).** In this path, established ML algorithms (ERT, kNN, LGBM, and NN as detailed in Sections 3.2.2.1, 3.2.1, 3.2.2.2, and 3.3.1) are applied directly to the input data set, resulting in what we term "raw" models.

- **Path 2 (bottom of Figure 31).** This approach utilizes the SMOGN data augmentation technique (see Section 3.5.1). Initially, the data is augmented, and then the same ML methods are applied to the augmented input data.

[1] Note that we do not employ this solution for non star-forming galaxies in Chapter 6. Moreover, we analyze these galaxies while accounting for observational effects in Section 6.3.

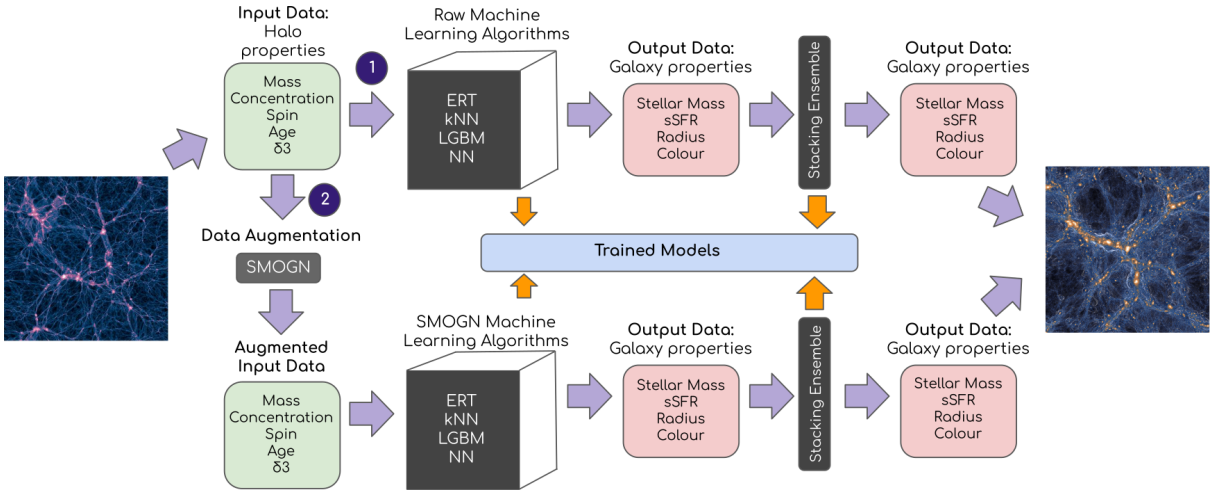

Figure 31 – **A schematic summary of the methodology followed by stacking raw and augmented models.** First, a data pre-selection is performed in order to generate the input data set. Once this initial catalog is ready, our method takes two different paths: in Path 1 (top), several ML algorithms are applied (ERT, κNN, LGBM, and NN) to the input data set; these are the "raw" models. A separate approach, which employs the SMOGN data augmentation technique, is shown in Path 2 (bottom). Subsequently, the same ML methods are applied to the SMOGN input data set. Both paths result in trained models for the galaxy properties. Finally, we have also implemented a stacking ML technique separately for each path, where all ML methods for the corresponding path are combined. The final output data set comprises our predictions for the galaxy properties under analysis. *Source:* Reference (65).

Both paths lead to trained models for the galaxy properties, which are provided as part of this approach. Additionally, we have implemented a stacking ML technique separately for each path, where all ML methods for the corresponding path are combined (see Section 3.5.2).

5.3.1 The SMOGN galaxy distributions

The SMOGN code was utilized by manually selecting regions of the distributions to be over- and under-sampled, based on visual inspection and considering the chosen proportion of normal and rare bins, as well as the specified number of k neighbors (see Subsection 3.5.1). Figure 32 illustrates the distributions of galaxy properties (stellar mass, sSFR, radius, and color) after applying the SMOGN data augmentation technique to the training and validation data sets. In essence, SMOGN enables us to enhance the statistics for underrepresented populations, such as those with high stellar mass, very low sSFR, very small and very large radius, and very blue color. It is important to note that these augmented distributions differ from the original ones because they are designed to compel the methods to learn how to predict properties in previously underrepresented regions in parameter space. Once the model is trained, the test data set (which is new to the machine and not augmented by SMOGN) is used to generate the expected results: the "complete" distribution, or something close to it.

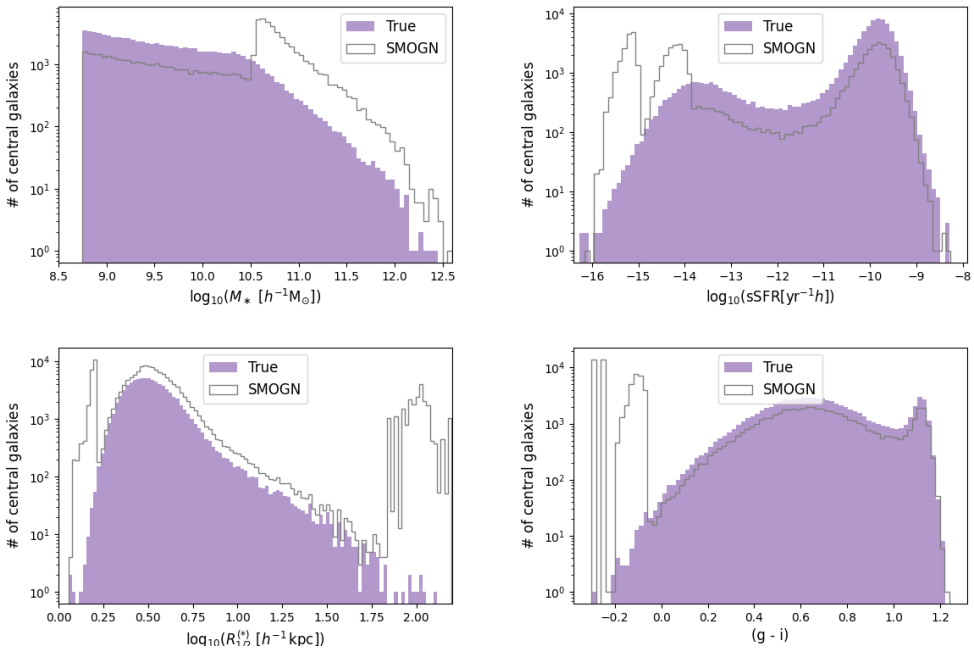

Figure 32 – **Augmented distributions by SMOGN.** A comparison of the histograms for the true and augmented SMOGN distributions for stellar mass, sSFR, radius, and color. *Source:* Reference (65).

5.3.2 Results

In this section, we present the main results of our analysis, focusing on our predictions of galaxy properties. We examine the frequency (see Section 5.3.2.1) and metric (in Section 5.3.2.2) performances, compare predicted values with true values (see Section 5.3.2.3) and analyze the distributions of halo-galaxy properties (in Section 5.3.2.4). Furthermore, in Section 5.3.2.6 we delve into our clustering measurements obtained through power spectrum analysis. Additionally, we explore the source of correlation between halo and galaxy property predictions by conducting a feature importance study, in Section 5.3.2.5.

5.3.2.1 Frequency performance

Our main goal is to recover some of the main galaxy properties, which means, in particular, to reproduce the frequencies (distributions) of each property. The true and predicted distributions are shown in Figure 33 for stellar mass (first row of plots), sSFR (second row), radius (third row), and color (fourth row). The true distributions are shown as the filled regions, while the distributions for the predictions using the ML methods are shown as lines. The first column shows the results for the "raw" ML models, which employ the original training set drawn from the TNG300 catalog. The second column corresponds to the SMOGN models, i.e., the ML models trained with the data which is augmented using the SMOGN technique. Finally, the third column shows the distributions for the stacked models (using all the different individual ML models) from the raw and SMOGN data sets.

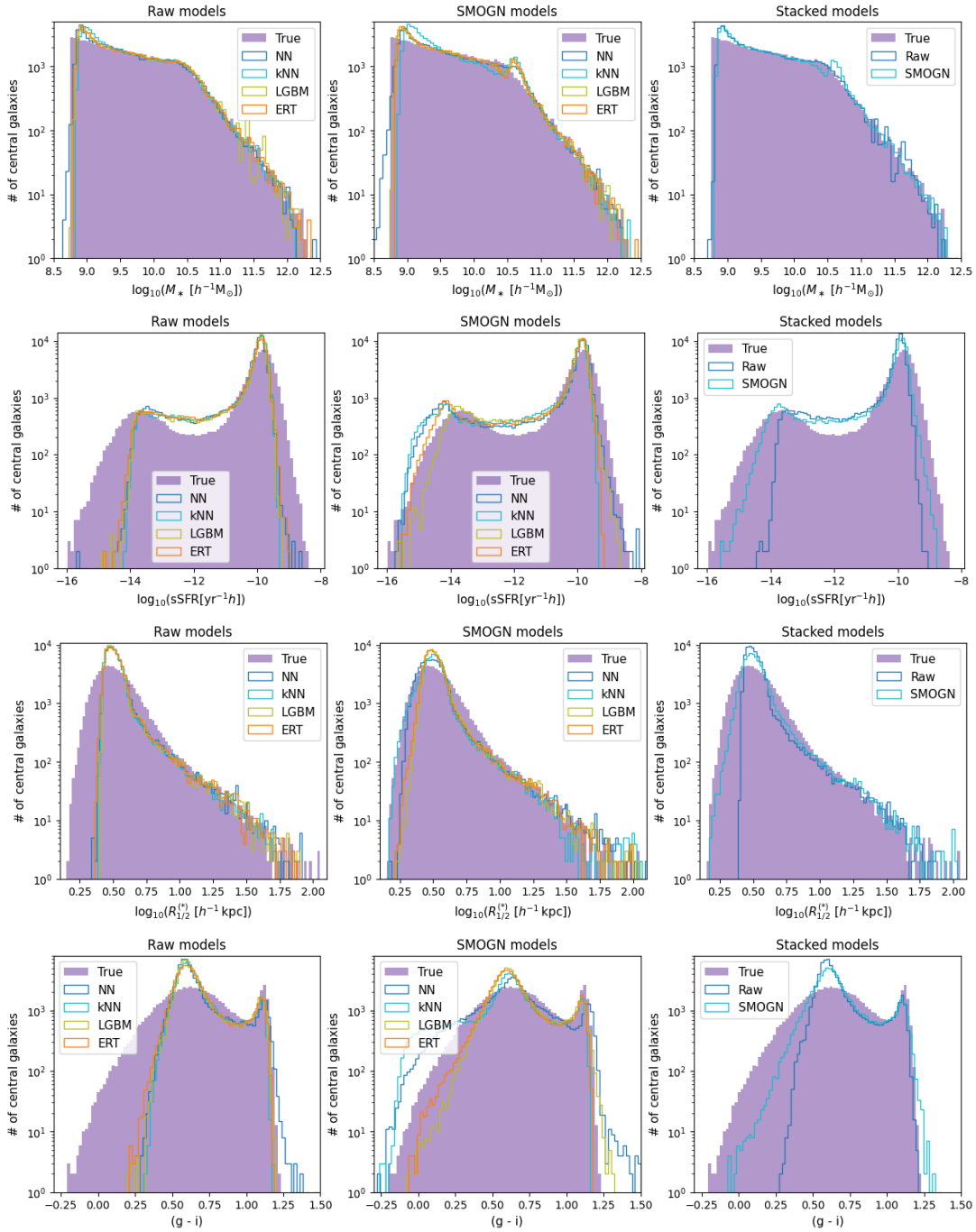

Figure 33 – **Frequency performance: Raw and SMOGN comparison.** A comparison of the histograms for the true and predicted galaxy properties for all ML models. Each row corresponds to a single galaxy property (stellar mass, sSFR, radius, and color). The first column corresponds to the "raw" ML models (i.e., ML models trained with the "raw" TNG300 data), the second column corresponds to the SMOGN models (i.e., the ML models trained with the augmented data, after using SMOGN), and the third column shows the distributions for the stacked models. *Source:* Reference (65).

The plots in Figure 33 clearly show that our machinery is capable of recovering the general distributions. As expected, stellar mass is the property that is better predicted by the models, with only some small deviations in the distributions. For the rest of the properties,

the performance of the machine worsens, indicating a more complex connection with the properties of the hosting halos (potentially due to information that we are not taking into account in our input sample).

The different individual algorithms provide similar results, mainly in the case of the raw models, but less so in the case of the SMOGN results, where the predicted distributions indicate that the tree methods still tend to privilege the peaks. This results in smaller improvements in the predictions at the tails of the distributions.

However, very large deviations are found when the raw and the augmented stacked models are compared. These differences are highlighted by the stacking ensemble models, which combine all the different ML models. The raw models are slightly less efficient at reproducing the scatter in the galaxy properties, especially towards the out skirts of the distributions, where underrepresented populations lie. They are, however, better at recovering the regions around the peaks of the distributions. Although the SMOGN methods are better at recovering the overall shapes of the distributions, we also notice the appearance of some artifacts, such as the small hump in the number of predicted galaxies around $\log_{10}(M_\star[h^{-1}M_\odot]) = 10.5$, which may be due to the SMOGN binning choices to under-sample low-mass objects and to over-sample large-mass ones (see the stellar mass histogram for SMOGN distribution in Figure 32).

5.3.2.2 Metric performances

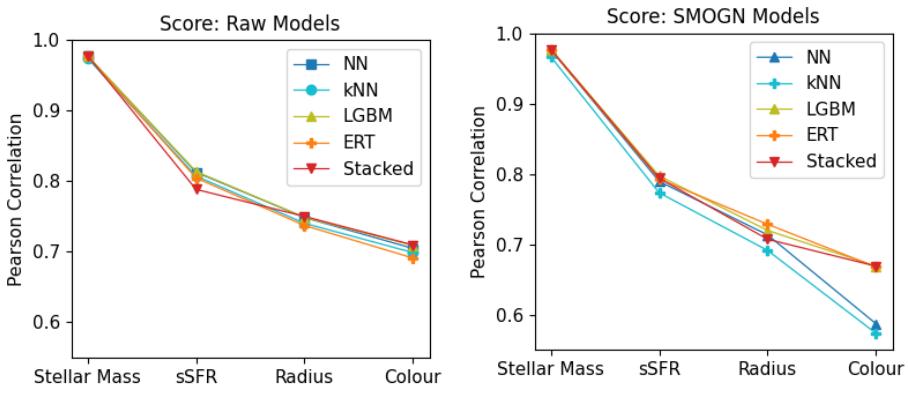

Figure 34 – **Pearson correlation coefficient comparison.** For each ML method and galaxy property. The left plot corresponds to the raw models, while the right plot displays the SMOGN models. Note that we have opted to connect the dots despite the fact that the properties in the x-axis are not correlated. This format is employed in order to facilitate the readability of the plots. *Source:* Reference (65).

In Figure 34, we show the PCC score (see Equation 5.3). This figure confirms that the best predicted galaxy property is stellar mass, both in terms of the raw, the SMOGN and the stacked models, reaching values of ~ 0.98. The Pearson correlation coefficient drops to ~ 0.80 for sSFR, $\sim 0.7 - 0.77$ for radius, and $\sim 0.57 - 0.71$ for color. The raw and SMOGN models perform, in general, similarly. The main difference resides in the lower scores for kNN and NN in the SMOGN models, particularly for color. This slightly worse performance of the SMOGN

models for some of the galaxy properties is directly connected to the augmentation technique itself. As mentioned before, the SMOGN method tends, by construction, to give greater weights to the tails of the distributions, especially in the case of color and radius.

Table 3 – **MSE and PCC scores.** Obtained for galaxy properties (in the test subset) for the raw and SMOGN models.

Property	Raw		SMOGN	
	MSE	**PCC**	**MSE**	**PCC**
Stellar mass	0.017	0.98	0.018	0.98
sSFR	0.691	0.79	0.747	0.79
Radius	0.012	0.75	0.014	0.71
Color	0.032	0.71	0.036	0.67

Together with the PCC results, we present in Table 3, the exact values for MSE and PCC, measured in the test subset for all galaxy properties, using the raw and SMOGN stacked models. Those results summarizes the advantages of using the stacked models compared to other works (e.g., PCC $\in [0.92, 0.957]$, from References (56, 184, 190) for stellar mass; MSE $= 0.126$ from Reference (56) for stellar mass; PCC $\in [0.745, 0.794]$, for SFR according to References (56, 184, 190)).

5.3.2.2.1 1D K-S test

The 1D K-S test is presented in Figure 35. It is again clear that stellar mass is the most easily predicted galaxy property, while the remaining properties are harder to determine just on the basis of the halo properties. Interestingly, the results of the K-S tests also indicate that the distributions of predicted radii and colors (and even sSFRs) for SMOGN reproduce better the true distributions. This result, again, reflects the philosophy behind the SMOGN technique. Finally, this clearly motivates the use of the stacked models (represented by the big markers on the right-hand side of each panel), as their performances are often very good in terms of recovering the true distributions. These results show that the stacked models are capable of providing a fair combination of the predictions of the different models.

5.3.2.3 Predicted versus True distributions

Figure 36 displays the scatter plots of the true *v.* the predicted values, for $30,000$ galaxies randomly chosen from the test sample. The color code represents the normalized density of objects. From these plots, it remains clear that the galaxy property best predicted is stellar mass, as the small scatter demonstrates. Our predictions become more uncertain for the rest of the galaxy properties. For sSFR, the models, particularly the SMOGN-augmented ones, perform relatively well towards the bulk of the distribution. However, despite our attempts to palliate the effect of the null-SFR values, objects with very small sSFR are still problematic for the

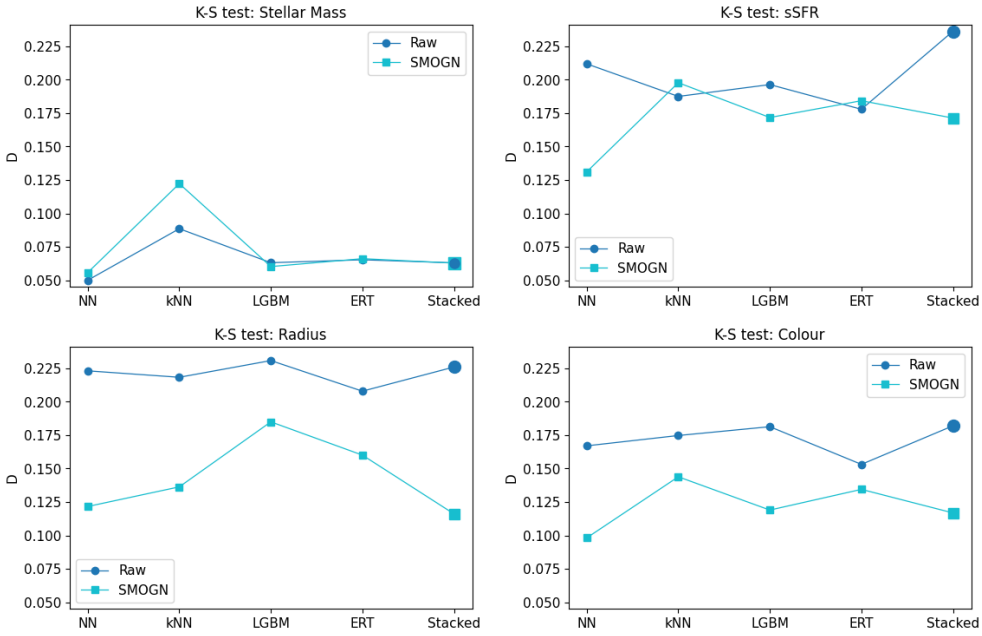

Figure 35 – **1D K-S test comparison.** Between the true and the predicted distributions, as described in Equation 5.4, for all galaxy properties and algorithms. Lower values correspond to better fits. Note that we have opted to connect the dots despite the fact that the properties in the x-axis are not correlated. This format is employed in this and other subsequent plots in order to facilitate the readability of the plots. *Source:* Reference (65).

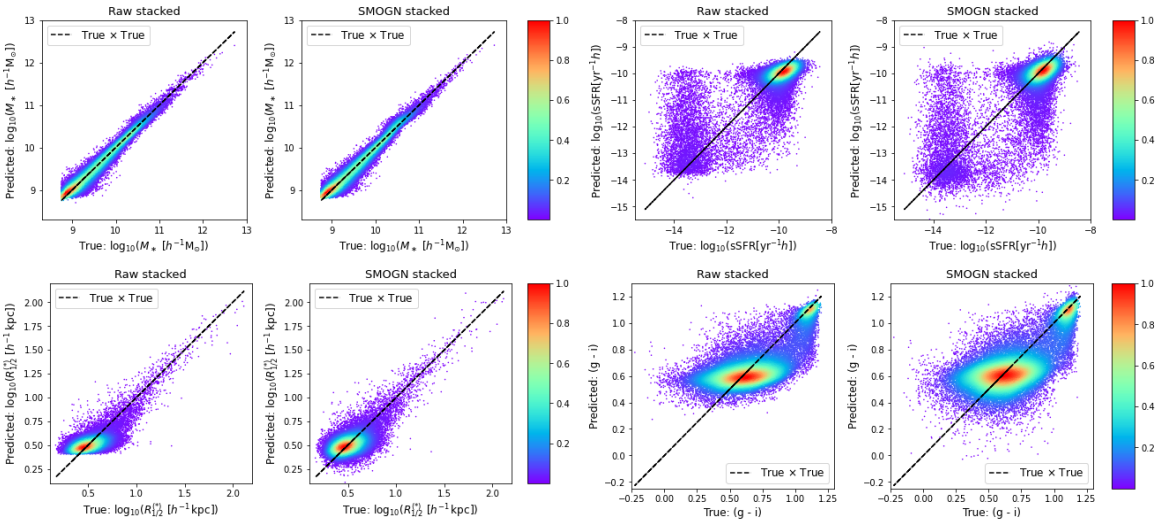

Figure 36 – **Predicted versus True distributions.** For stellar mass, sSFR, radius, and color, for raw (left) and SMOGN (right) stacked models. The color code indicate the normalized density of objects. *Source:* Reference (65).

machine. In the case of galaxy radius, our predictions for the largest objects are good and unbiased. For the smaller, more common objects, the raw model predicts a distribution that is tilted with respect to the real one. This effect is due to the fact that the machine predicts a narrower range of values for this property. Something similar happens for galaxy color, where, again, the bimodality is well reproduced, but the predicted blue cloud is severely tilted as

compared to the real data (an effect that is not as strong for the red sequence).

A very important advantage of SMOGN seems to emerge here. SMOGN tends to rectify this problem, reducing the tilt in the distributions. This improvement, which is still not complete, does suggest that using the augmented data set allows the machine to predict a wider range of output values. The effect is particularly evident for color and radius, where the raw models are unable to predict any galaxies with $(g - i) < 0.3$ and radii lower than $0.375h^{-1}$ Mpc, whereas the SMOGN results do.

5.3.2.4 Halo-galaxy property distributions and the 2D K-S test

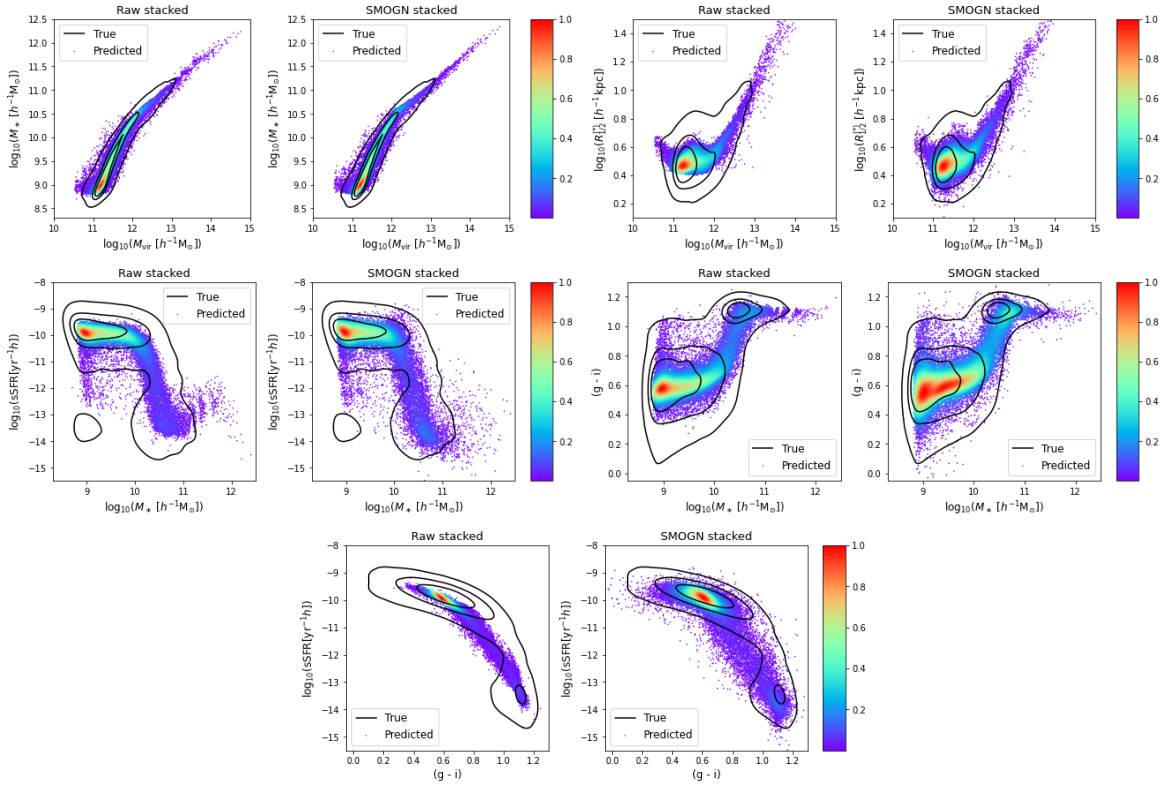

Figure 37 – **Halo-galaxy property distributions.** We present: stellar mass *v.* halo mass, radius *v.* halo mass, sSFR *v.* stellar mass, color *v.* stellar mass, and sSFR *v.* color. For all relations, the stacked raw (left) and SMOGN (right) models are compared. The color code represents the normalized density of objects and the true distributions are shown in black contours. *Source:* Reference (65).

It is important to analyze the ability of the method to reproduce the relations between different galaxy and halo properties (which can be directly seen as the halo-galaxy connection). Figure 37 shows the relations: stellar mass *v.* halo mass, radius *v.* halo mass, sSFR *v.* stellar mass, color *v.* stellar mass, and sSFR *v.* color (in the same format of Figure 36). Generally speaking, we have encouraging results, demonstrating that the machine produces fairly realistic relations between properties. One of the main problems to overcome, as this figure illustrates, is the scatter in these relations. By construction, ML models tend to concentrate on the bulk of the

distributions, which hinders the prediction of scatter. A clear example of this is the mass–size relation. Again, SMOGN works in the right direction, increasing the scatter in the relations.

Table 4 – **2D K-S test comparison.** For the stacked raw and SMOGN models for the joint halo-galaxy distributions.

Joint property	D (raw stacked)	D (SMOGN stacked)
Stellar Mass *v.* Halo Mass	0.065683	0.065417
Radius *v.* Halo Mass	0.228733	0.116850
sSFR *v.* Stellar Mass	0.272300	0.208417
Color *v.* Stellar Mass	0.224533	0.164350
sSFR *v.* Color	0.327667	0.282467

In order to quantify the results of Figure 37, we have computed the 2D K-S test between the true and predicted joint distributions. The results of this test are presented in Table 4. For all pairs of properties, the distances between the cumulative distributions are close to zero, and here again the best result is for stellar mass *v.* halo mass. The decrease of the value for each relation from raw to SMOGN models is remarkable, reaching its highest difference for radius *v.* halo mass, followed by color *v.* stellar mass.

5.3.2.5 Feature Importance Analysis

The ultimate goal of our analysis is to establish relations that allow us to shed light onto the intricacies of the halo-galaxy connection. One of the ways to address this aspect and to gain some insight into the inner workings of the ML methods is to analyze the weights given to each feature (i.e., halo property) which contributes to producing the desired output.

Although tree-based methods have been employed in our work, we are not utilizing their inherent mechanisms to compute this statistic. This decision stems from our objective of seeking a method applicable to all the ML methods under consideration, including kNN and NNs. Therefore, to compare the relative weights of the input features across different ML methods, we have opted to compute the *permutation feature importance*, as detailed in Section 3.5.3) (186). For this purpose, we utilize Eli5 to perform this computation. For clarity, all our results have been converted to percentage of the feature importance for halo mass, which is the predominant parameter for determining all galaxy properties.

Figure 38 shows the importance of the halo features for each individual ML model. Here, a hierarchy emerges: on the one hand, the halo mass is clearly the most important feature for the prediction of all the galaxy properties, as expected. On the other hand, the environmental property δ_3 had an almost negligible level of importance for all the galaxy properties. Somewhere in the middle, age turns out to be quite important for color and sSFR, but less so for stellar mass and radius. Galaxy radius is perhaps the most interesting case, where

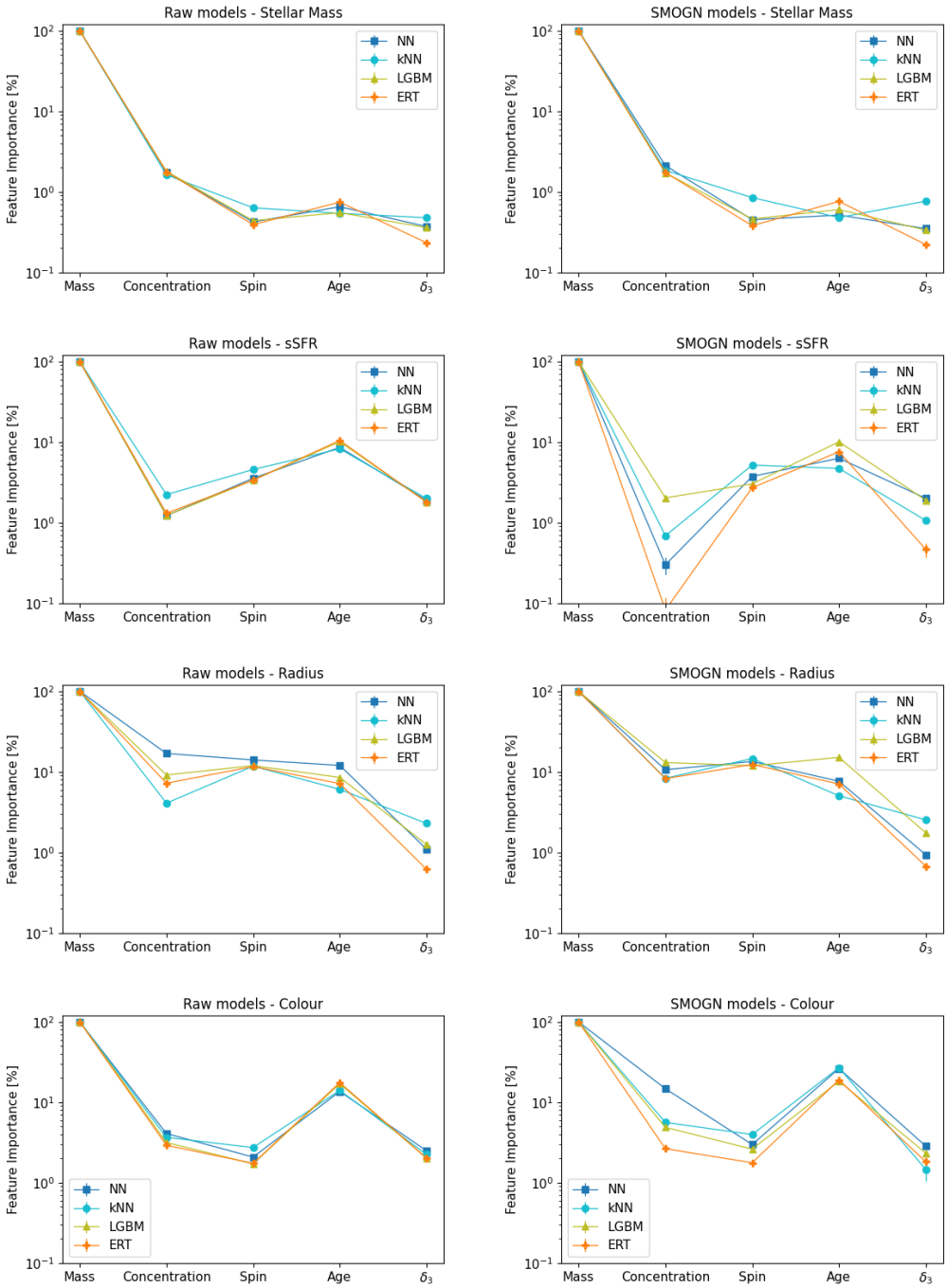

Figure 38 – **Permutation feature importance comparison for the different ML models.** For raw models (left panels) and SMOGN models (right panels). Each row corresponds to the predictions for stellar mass, sSFR, radius, and color, respectively. This plot shows the weights given to each individual halo property (mass, concentration, spin, age, and δ_3), normalized with respect to the feature importance of the halo mass parameter. *Source:* Reference (65).

a number of halo features appear in a less hierarchical way, with concentration, spin, and age contributing at about the same level ($\sim 10\%$ compared to halo mass).

We can also apply the idea of feature importance to the stacked models, measuring the

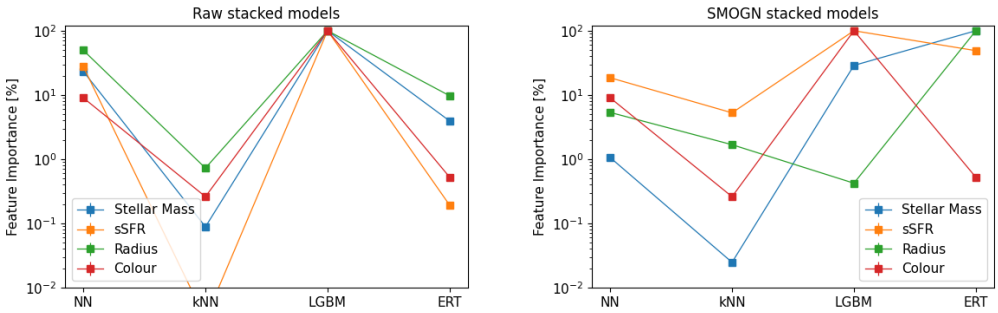

Figure 39 – **Permutation feature importance comparison for the stacked raw and SMOGN models.** For each of the predicted galaxy properties (stellar mass, sSFR, radius, and color). This plot shows the weights given to each individual ML model (NN, kNN, LGBM, and ERT) in the stacking procedure, normalized with respect to the feature importance of the LGBM model. *Source:* Reference (65).

level of importance of the different ML algorithms. In that case, the features under analysis are the predictions of the individual ML models. The results have been normalized to the LGBM predictions, since that is the predominant model for determining the stacked predictions for all galaxy properties (the same way that halo mass was the predominant feature before). The results are shown in Figure 39, where the upper and lower panels correspond to the raw and SMOGN models, respectively. Recall that, for the raw models, the best predictions are obtained using the LGBM and NN methods, while the kNN and ERT methods return worse predictions, depending on the galaxy property. The SMOGN methods behave differently, with LGBM and ERT yielding the best results.

5.3.2.6 Power Spectrum

An important test to our methods is the clustering properties of the galaxies whose properties we are trying to predict. By splitting those galaxies in two populations according to those properties, and then computing the clustering of each population, we can check whether our predictions are able to separate the galaxies correctly, according to the types of halos that they inhabit. Furthermore, given that we have exactly the same DM halos, by splitting the galaxy populations both in terms of their predicted properties, as well as their true properties, we can assess some of the systematics that arise in the bias of those populations as a result of our imperfect predictions, in a way that is protected against cosmic variance.

We have split the galaxies according to each property (stellar mass, sSFR, radius and color) in two bins each, with bin edges and central values listed in Table 5. For the true galaxies, we use their positions from the TNG300 catalog, while for the ML predictions we use the positions of their hosting halos. All the spectra were then measured for the entire TNG300 box, computing the FKP spectra (see Sections 2.3.1 and 2.3.3, Equation 2.27) using the Python package NbodyKit (96). Because we have only one single IllustrisTNG box, the uncertainties of the spectrum on each of the Fourier bins (bandpowers) k_i, for each tracer α, $\sigma_{P_{\alpha,i}}$, were

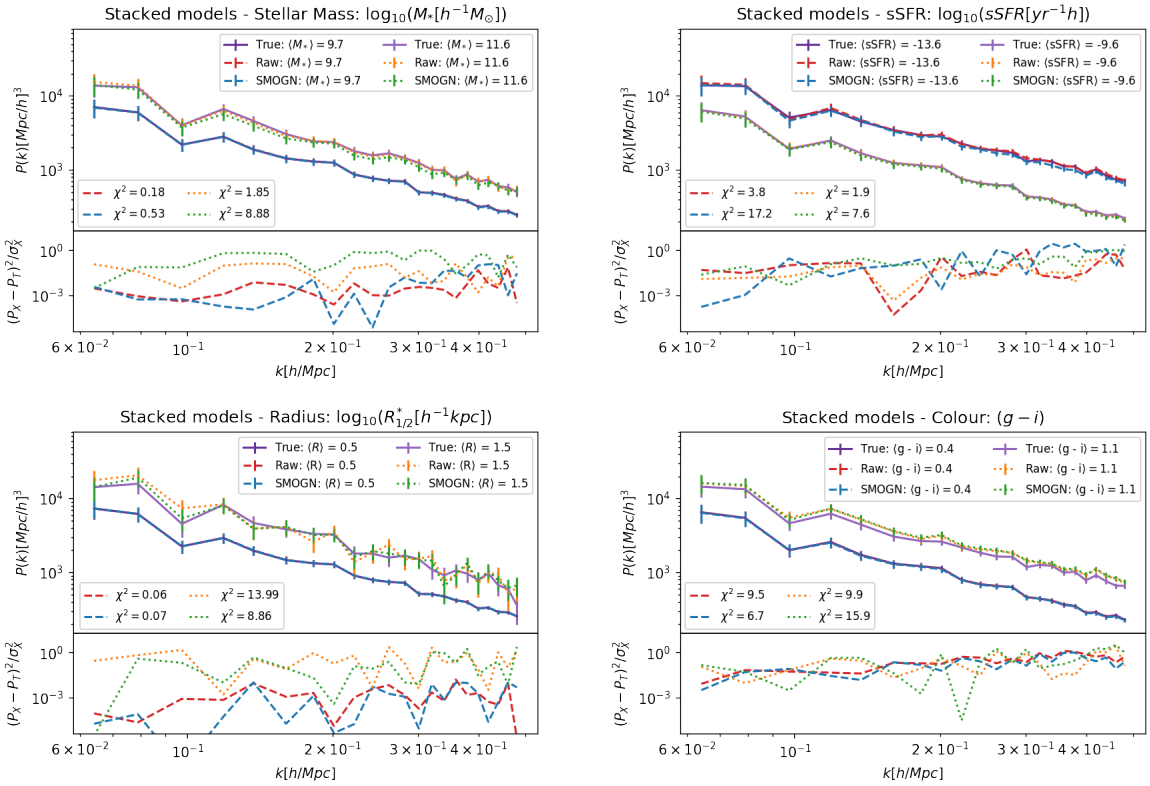

Figure 40 – **Power spectra comparison.** For the raw and SMOGN stacked models compared with the true data set. In each panel, a different galaxy property, split in two bins, is analyzed (see text). The residuals are provided in the subplots. *Source:* Reference (65).

Table 5 – **Bin edges and central values for the subsets considered for the computation of the power spectrum.**

Property	Bin edges	Central values
$\log_{10}(M_*\ [h^{-1}\mathrm{M}_\odot]\ )$	$[8.8, 10.6, 12.7]$	$[9.7, 11.6]$
$\log_{10}(\mathrm{sSFR}[\mathrm{yr}^{-1}h])$	$[-16.3, -11.0, -8.3]$	$[-13.6, -9.6]$
$\log_{10}(R^{(*)}_{1/2}\ [h^{-1}\,\mathrm{kpc}])$	$[0.1, 0.9, 2.1]$	$[0.5, 1.5]$
$(g-i)$	$[-0.23, 1.0, 1.2]$	$[0.4, 1.1]$

assumed to be given by:

$$\frac{\sigma^2_{P_{\alpha,i}}}{P^2_{\alpha,i}} = \frac{2}{\tilde{V}_i V}\left(\frac{1+\bar{n}_\alpha P_{\alpha,i}}{\bar{n}_\alpha P_{\alpha,i}}\right)^2 , \tag{5.5}$$

where $\tilde{V}_i = 4\pi k_i^2 \Delta k/(2\pi)^3$ is the volume of the Fourier bin, with Δk representing the width of the bandpower, V is the volume of the catalog, and $\bar{n}_\alpha$ is the mean number density of the tracer α (which here stands for the two bins in galaxy properties).

The results for the power spectrum are shown in the four panels of Figure 40, which, from upper left to lower right, correspond respectively to binning the galaxies in stellar mass, sSFR, radius, and color. The legends in the upper right corner show the central values of the

bins of each galaxy property, for each bin and method – which, as can be seen, are identical for all methods. The legends in the lower left corners indicate the χ^2 values for the fit of the spectra of the predicted versus the true galaxies.

For each plot in Figure 40 we also show the residuals:

$$\frac{\left[P^{Pred}_{\alpha,i}(k) - P^{True}_{\alpha,i}(k)\right]^2}{\left[\sigma^{Pred}_{P_{\alpha,i}}\right]^2}\ , \tag{5.6}$$

where $Pred$ is the predicted and $True$ is the true power spectrum (i.e., the power spectrum that results from using the original galaxy property and position in the TNG300 box). For stellar mass, the residuals range between 10^{-5} and 0.12, for the less massive bin, and $1.6 \cdot 10^{-3}$ and 1 for the most massive interval. In the case of sSFR, the residuals range between $5.5 \cdot 10^{-5}$ and 2.9 for the first (low sSFR) bin, and $5 \cdot 10^{-4} - 1.0$ for the second subset (high sSFR). Similar results are obtained for galaxy size and color. For the former, ranges of $[2 \cdot 10^{-8}, 0.02]$ and $[3 \cdot 10^{-6}, 2.5]$ are obtained for the residuals in each bin, respectively. Finally, the residuals in the power spectrum for galaxy color are $[0.009, 1.3]$ and $[3 \cdot 10^{-5}, 3.3]$, for the blue and red subsets, respectively.

Particularly, the residuals follow the same trend for sSFR and color, and behaves differently for stellar mass and radius. In the former case, the comparison happens because both bins either have their mean values (for the raw and SMOGN predictions) close to the true spectra, or because the dispersion σ is higher enough (specifically in the case of the red color bin). In the latter case, it is evident that the bins with a low number of objects (higher stellar masses and higher radii) have higher values for the residuals. Some trends, such as the fact that the residuals increase with k for all four predicted properties, show that our predictions are more accurate on larger scales. This can be both because of shot noise (which affects more the small scales), and also because it is harder for the predictions to match precisely the local environments of those galaxies and halos. Another point to consider is that binning the galaxy populations sometimes leads to samples with very different sizes, which also affects the residuals.

When adding up the residual for all the values of k, we obtain the χ^2 associated to each power spectrum. Since we have 22 bins of k, the χ^2 per degree of freedom is significantly smaller than 1 in all cases, which is indicative of an excellent agreement. It is noteworthy that the SMOGN stacked models perform slightly worse for stellar mass and sSFR, compared with the raw stacked models, but they do better for radius and color.

5.3.3 Discussion and conclusions

The predictive power of ML techniques can be harnessed to reproduce the hidden intricacies of the halo-galaxy connections. The main goal in this field is to establish relations between the properties of galaxies and the properties of their hosting halos, in the cosmological context of the large-scale structures of the Universe. This problem can be treated in ML in

terms of an *input* data set (halo properties), which is known a priori, and an *output* data set, corresponding to the galaxy properties that we attempt to predict (53,56,58,61,184,190,274,275).

We have selected four different ML algorithms (NN, kNN, LGBM, ERT), as well as the combination of their predictions (the stacked models), and evaluated their ability to predict stellar mass, color, sSFR, and half-mass radius for central galaxies in the TNG300 hydrodynamical simulation. In addiction, we have employed a data augmentation technique called SMOGN for the first time in the context of the halo-galaxy connection field.

Our set of halo properties includes halo mass, age, concentration, spin, and overdensity around halos. Overall, our findings are consistent with previous results in the literature, with stellar mass being the most accurately predicted property, with a PCC of ~ 0.98 (previously reported values are typically $0.92 - 0.957$, see References (56, 184, 190)). The second best-predicted property is sSFR, with a correlation coefficient of ~ 0.8 (previously, $0.745 - 0.794$, see References (56, 184, 190)). For size and color we obtain coefficients in the range $0.7 - 0.8$ and $0.59 - 0.71$, respectively. A similar hierarchy for the predictive power of our ML methods is obtained when other performance estimators such as the K-S test are employed.

The aforementioned improvements are primarily due to the use of both the stacked models and the SMOGN technique. The stacked models perform a linear combination of the different ML predictions for each galaxy property. The SMOGN method, on the other hand, alleviates the problem of imbalanced data sets, by statistically compensating the lower-populated regions in parameter space. Note that ML methods tend to focus, by construction, on reproducing the better represented data. Needless to say, in the context of the halo-galaxy connection, there is significant value in the sparse (rare) objects. We have shown in a quantitative manner that the use of SMOGN has several advantages. First, it helps predicting galaxies in the tails of the distributions (this can be seen from the distribution histogram or from the small D-values obtained from a K-S test: e.g., $D \lesssim 0.175$ for sSFR and $D \lesssim 0.125$ for radius and color). Second, it tends to rectify the tilted real v. predicted distributions obtained using the raw models (without augmentation), by expanding the predicted range of values.

The above advantages are also noticeable when the joint distributions of galaxy/halo properties are analyzed. We have demonstrated that we are capable of reproducing very well the shape of several important relations, such as the stellar/halo mass relation (SHMR) or the galaxy size–halo mass relation, to name but a few. Here, SMOGN proved once again to be helpful, particularly in terms of reproducing the overall scatter in the relations. We have used the 2D K-S test to quantify the accuracy of our predictions, both for the raw and for the SMOGN stacked models. To give an example, the use of the data augmentation technique improves our predictions by $\sim 49\%$ and $\sim 27\%$ (as compared to the raw models) for the galaxy radius–halo mass relation and the color–stellar mass relation, respectively. In conclusion, the results presented in this work clearly show the potential of SMOGN in the context of the halo-galaxy connection.

In terms of the physical implications of our results, the most important aspect is the quantification of the *feature importance*, i.e., the contribution of each halo property to the prediction of each galaxy property. The validity of this analysis is of course due to the high consistency across the different models. As expected, stellar mass seems to be almost completely determined by halo mass, whereas the inclusion of halo age is necessary to predict both sSFR and color. Maybe the most interesting case is again galaxy size, which, within the uncertainties of our analysis, seems to be primarily determined by halo mass (this makes sense due to the mass–size relation). The contribution of other properties such as spin, concentration or age seem to be equally relevant and significant, but it is important to bear in mind that our prediction for galaxy radius is still not optimal. These results seem to be connected with the shape of the (halo/stellar) mass–size relation (see, e.g., Reference (302)): at the high-mass end, central galaxy size is proportional to halo mass, but the relation is basically flat at the low-mass end.

The study of the halo–galaxy connection would be incomplete if the relations between the properties of halos/galaxies and their spatial distribution in the large-scale structure were not taken into account. In the last part of this work we show that the clustering of our predicted central galaxies, measured in terms of their power spectra, reproduces that of the true sample with a high level of accuracy: $0.05 - 7.6\%$, $\chi^2 = 0.18 - 8.88$ for stellar mass; $2.0 - 5.1\%$, $\chi^2 = 1.9 - 7.6$ for sSFR; $0.12 - 11.7\%$, $\chi^2 = 0.06 - 13.99$ for radius; $4.4 - 7.3\%$, $\chi^2 = 6.7 - 15.9$ for color. Importantly, this good agreement is obtained for multiple subsets defined in terms of the aforementioned galaxy properties. Despite this performance, some subsets display a few percent bias (difference) in the amplitude of the spectrum. As an example, the high-mass subpopulation, $\log_{10}$ ($\mathrm{M}_*[h^{-1}\mathrm{M}_\odot]) > 10.6$, and the high-sSFR subpopulation, $\log_{10}(\mathrm{sSFR}[\mathrm{yr}^{-1}h]) > -11$, are predicted to have slightly smaller bias than the real TNG300 galaxies. On the other hand, the subpopulation with large radius, $\log_{10}(R^{(*)}_{1/2}[h^{-1}\,\mathrm{kpc}]) > 0.9$, and that with bluer colors, $(g-i) < 1$, show some scatter, but no significant bias, especially on small scales ($k \gtrsim 0.1h$ Mpc^{-1}), which may point towards either hidden variables that correlate with these properties, or a larger role of stochasticity. In terms of the clustering properties of IllustrisTNG galaxies, one interesting aspect that merits further investigation, particularly if larger boxes become available, is whether clustering can be reproduced *at fixed halo mass*.

The results presented in this work come with another important realization. Even though we are equipped with a powerful ML machinery and the SMOGN augmentation technique, the accuracy in the predictions for galaxy radius, sSFR, and color is still not comparable to that of stellar mass. In Figure 41, we show the effect of considering galaxy properties as input features (using only NN), i.e., we use all halo and galaxy properties (except for the one under analysis) to predict stellar mass, sSFR, radius, and color. This exercise is reassuring in terms of the robustness of our methodology, since our predictions, in most cases, improve significantly (both qualitatively and quantitatively). That is the case for sSFR and color (stellar mass was already very well reproduced), which is of course expected due to their correlation.

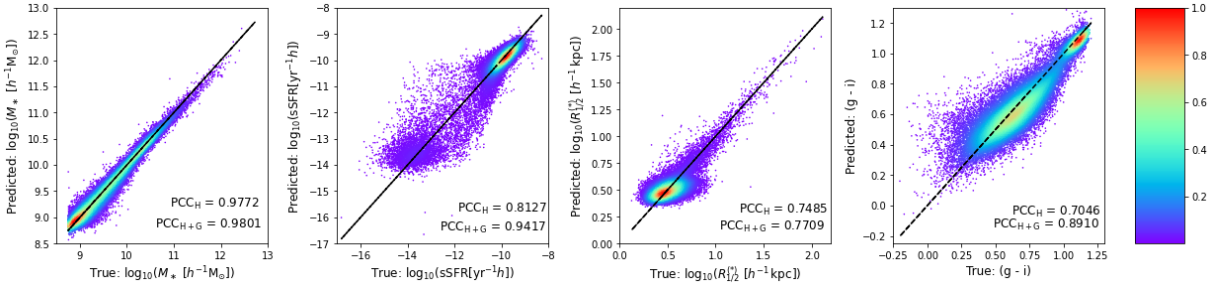

Figure 41 – **Predicted versus true distribution using halo and galaxy information.** For stellar mass, sSFR, radius, and color for NN models produced using all halo and galaxy properties (except for the one under analysis). Each point represents a central galaxy and the color bar corresponds to the normalized density of objects in each region. *Source:* Reference (65).

Figure 41, however, illustrates some of the challenges: even with the aid of galaxy properties, there is a substantial level of scatter which cannot be overcome with this set of properties alone. For sSFR, the scatter becomes very small for the bulk of the distribution, but there are still problems at the low-sSFR range–a testament to the challenge of dealing with extremely low-SFR objects in TNG300. For color, conversely, the scatter is larger in the blue cloud than in the red sequence. Finally, particularly striking is the effect on galaxy size, where little or no improvement is observed in the performance scores, nor in the visual appearance of the scatter distribution.

An interesting point of debate is whether the predictions for these properties can be significantly improved by including any additional halo or environmental property (or information on the assembly history, such as the number of major mergers). Equivalently, one can ask to which extent the problem is dominated by the intrinsic stochasticity of the galaxy formation process. In this sense, we have tweaked our definition of environmental overdensity, by varying the threshold scale. None of these tests provided significant improvements in our performance scores with respect to the basic set of halo properties, so we have opted to stick to our fiducial configuration for simplicity. However, as shown in References (66, 277), the use of merger trees have already obtained better results than what we show here.

In the same context, the fact that we are able to predict the TNG300 clustering with precision, even when the sample is split in multiple ways, serves as a motivation to explore in more detail the related effect of *galaxy assembly bias*. Galaxy assembly bias measures the dependence of the properties and clustering of galaxies on halo properties beyond halo mass (see References (293, 303)), and can be a useful tool both to test our results as well as a property that can be directly predicted using ML techniques.

5.4 Converting regression to classification

The methodology employed in this section involves converting the regression problem discussed in Section 5.3 into a classification problem, as outlined Section 3.4.2. We utilize the same data set and data preprocessing techniques detailed in Section 5.1. This approach has its complete version in Reference (69). In this section, we provide a summary of some of the key achievements, comparing them with the findings presented in the previous section (See Section 5.3).

5.4.1 Methodology

To start, we train four models to predict each galaxy property individually as univariate distributions. This means we have separate models to predict $P(M_*)$, $P(g-i)$, $P(\text{sSFR})$, $P(R^{(*)}_{1/2})$. While this approach is sufficient to recover the overall distribution $P(Y)$ for a given sample, it does not guarantee, *a priori*, that the joint distributions are well reproduced.

Therefore, we proceed to predict pairs of properties, namely $P(M_*, g-i)$, $P(M_*, \text{sSFR})$, $P(g-i, \text{sSFR})$, and $P(R^{(*)}_{1/2}, M_*)$. This strategy is similar to the univariate $P(Y)$ case: we make a grid in the $\{Y_1, Y_2\}$ subspace so that the output corresponds to pixels in this grid.

For all the results shown here, we set $K = 50$ classes for each one of the central galaxy properties, with equally spaced bins. For example, for stellar mass, this corresponds to bins of 0.085 dex. It is important to note that this choice of binning is arbitrary. We have tried different numbers of bins, finding similar results in terms of the recovery of the distributions.

5.4.2 Results

In this section we present some of the results we have achieved in Reference (69), mostly based in the comparison with Reference (65) (see Section 5.3). We start presenting the comparison for the distribution of halo and galaxy properties, in Section 5.4.2.1. In Section 5.4.2.2, we present the K-S test for the predicted individual and joint galaxy properties. In order to be able to compare the probabilistic results with one-single value estimations, we compute the PCC score in Section 5.4.2.3.

5.4.2.1 Distribution of halo-galaxy properties

In Figure 42 we present the distributions of the galaxy properties for the test set. The first column is the truth table, the TNG300 catalog. The second column is the NNCLASS prediction of univariate distributions, i.e., galaxy properties predicted independently. With the univariate distributions we can compute the joint distributions as $P(Y_1) \otimes P(Y_2)$, which are shown in the heatmap diagrams. The third column is the NNCLASS prediction for the joint distributions $P(Y_1, Y_2)$, which can be integrated to recover the univariate distributions $P(Y)$ shown in the

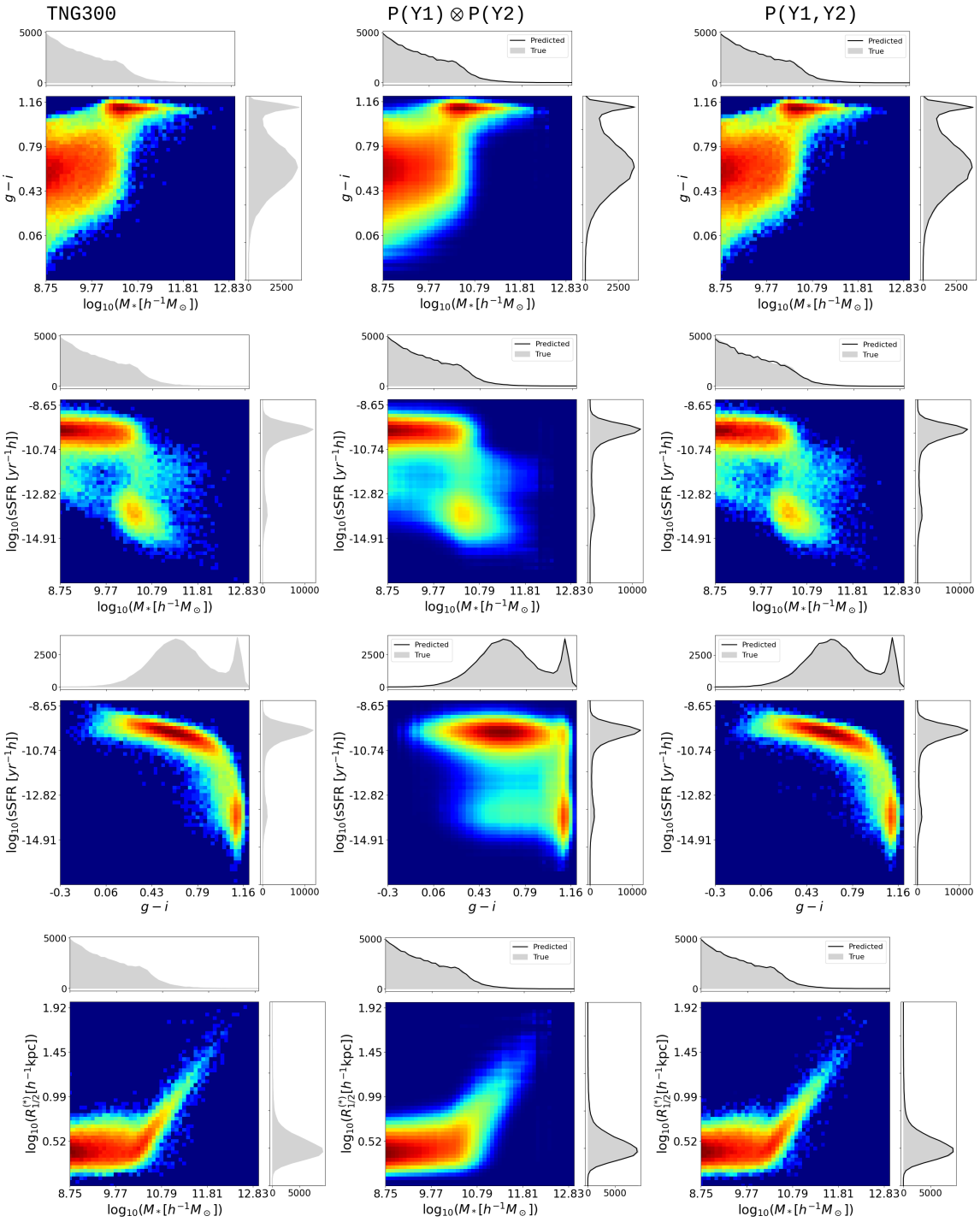

Figure 42 – **Distributions of galaxy properties.** From top to bottom: color $v.$ stellar mass, sSFR $v.$ stellar mass, sSFR $v.$ color, and radius $v.$ stellar mass. The first column shows the true distributions from TNG300. The second column shows the distributions computed from the univariate distributions as predicted by NNClass–i.e., predicted independently from each other. The third column shows the joint distributions as predicted by NNClass. The gray shaded regions in the marginal plots correspond to the TNG300 distributions, while the black solid lines correspond to the NNClass predictions. The univariate distributions shown in the third column plots were computed by marginalizing the joint distributions. *Source:* Reference (69).

marginal plots from the third column, i.e.:

$$P(Y_i) = \int P(Y_i, Y_j) dY_j. \tag{5.7}$$

The univariate distributions predicted by NNClass, shown in black solid lines in the second-column plots of Figure 42, are in excellent agreement with the true distributions from TNG300, shown in gray shaded regions. They also reproduce fairly well the joint distributions $P(Y_1) \otimes P(Y_2)$ for most cases. The $P(g-i) \otimes P(\text{sSFR})$ joint distribution, however, fails to reproduce the shape of the distribution for redder colors and lower sSFRs. According to this prediction, red galaxies could have virtually any value of sSFR, while what we actually observe in TNG300 is that as galaxies move from the blue to the red the peak, their sSFRs decrease. This important feature is recovered when NNClass is trained to predict $P(g-i, \text{sSFR})$ jointly (third column in Figure 42).

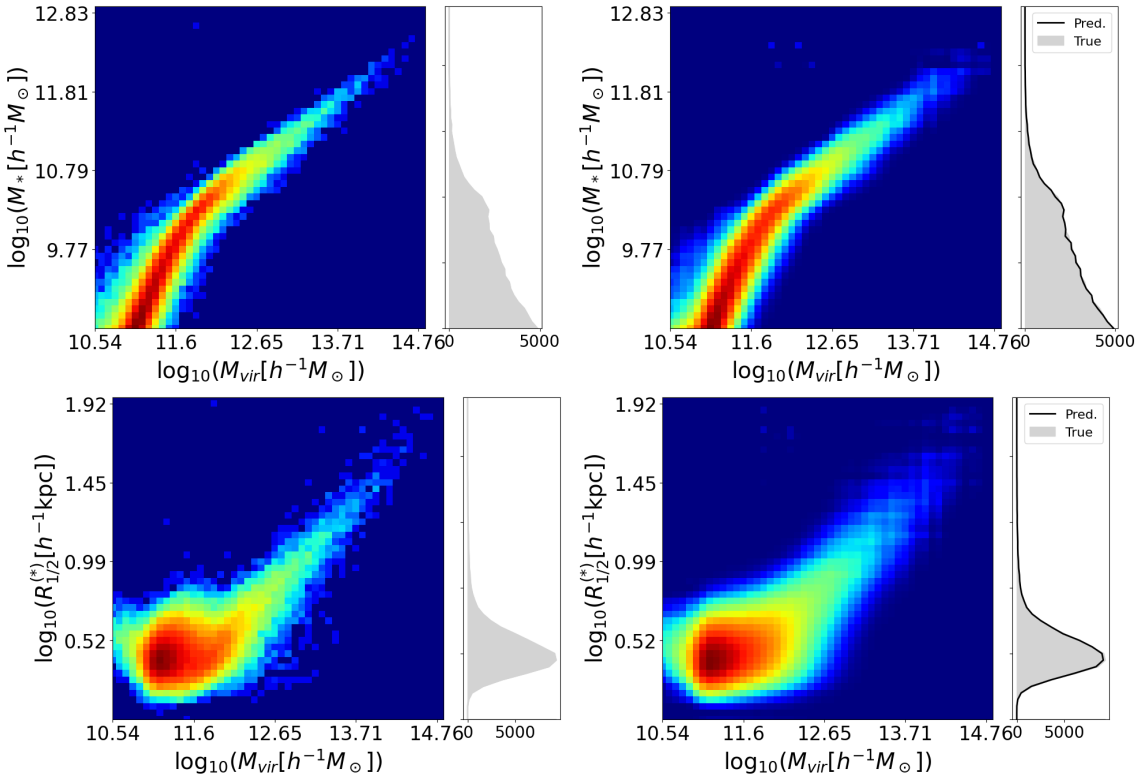

Figure 43 – **Distribution of halo-galaxy properties.** Stellar-to-halo mass relation (top) and galaxy size–halo mass relation (bottom) from the TNG300 catalog (left panels) and from NNClass predictions (right panels). *Source:* Reference (69).

As a complementary analysis, in Figure 43 we show: the stellar-to-halo mass relation, and the galaxy size–halo mass relation obtained with TNG300 and with $P(M_*)$ and $P(R_{1/2}^{(*)})$ predicted by NNClass.

The above result indicates that our input halo properties alone are unable to predict accurately the correlations between color and sSFR. The model would need additional features in order to capture this relation (as observed in Section 5.3.3 or directly in Reference (65)). It is

interesting, however, that we can overcome this limitation by predicting the joint distribution directly using only the presented halo properties. This exercise indicates that, in order to robustly assign galaxies to halos, with all the properties consistently correlated, the properties should be predicted together (overall distributions which are way better than the ones presented in Section 5.3.2.4 (65)). Note that, in principle, one could define galaxy populations based on as many parameters as wished. Therefore, in the most general case, we would have an N-dimensional distribution associated to each host halo.

5.4.2.2 K-S test for galaxy predictions

Table 6 – **K-S test values for univariate (1D) and joint (2D) distributions computed with the NNs and the baseline models.**

1D KS	$P(Y)$	Raw	SMOGN	**2D KS**	$P(Y_1) \otimes P(Y_2)$	$P(Y_1, Y_2)$	Raw	SMOGN
$P(M_*)$	0.002	0.064	0.064	$P(M_*, g-i)$	0.010	0.005	0.183	0.163
$P(g-i)$	0.004	0.181	0.116	$P(M_*, \text{sSFR})$	0.012	0.009	0.253	0.209
$P(\text{sSFR})$	0.004	0.213	0.168	$P(g-i, \text{sSFR})$	0.110	0.009	0.266	0.176
$P(R_{1/2}^{(*)})$	0.009	0.217	0.110	$P(M_*, R_{1/2}^{(*)})$	0.015	0.007	0.217	0.150
				$P(M_{\text{vir}}, M_*)$	0.008	–	0.064	0.064
				$P(M_{\text{vir}}, R_{1/2}^{(*)})$	0.012	–	0.217	0.110

Figures 42 and 43 allow for a visual inspection of the results. In order to quantify the similarity between the distributions, we have performed the K-S test (see Section 5.2). The results are shown in Table 6. For comparison, we also show the values obtained with our baseline models, Raw and SMOGN, from Section 5.3 (65). Once again, we see that for most cases the independent prediction of univariate distributions reproduce fairly well the joint distributions, except for color and sSFR. In all cases, NNClass provides significantly lower values as compared to Raw and SMOGN.

5.4.2.3 Single Value Estimation

We can also discuss the results of NNClass in terms of single-point estimation scores. Since we do not have a single value associated to each data set instance, but a distribution, one can sample several times from this distribution in order to estimate the most probable value, and compute single-point estimation metrics with it. In practice, we take the average of the number of realizations ($r \in [1, 42]$) of each predicted galaxy property, and calculate the PCC between the true and estimated values (see Equation 5.3).

Figure 44 shows the PCC score as a function of the number of realizations and also the values of the baseline models for the four galaxy properties. We sample from univariate distributions $P(Y)$ instead of joint distributions. NNClass provides results comparable to the single-point estimators Raw and SMOGN as the number of realizations increases, which indicates that NNClass are also good maximum likelihood estimators.

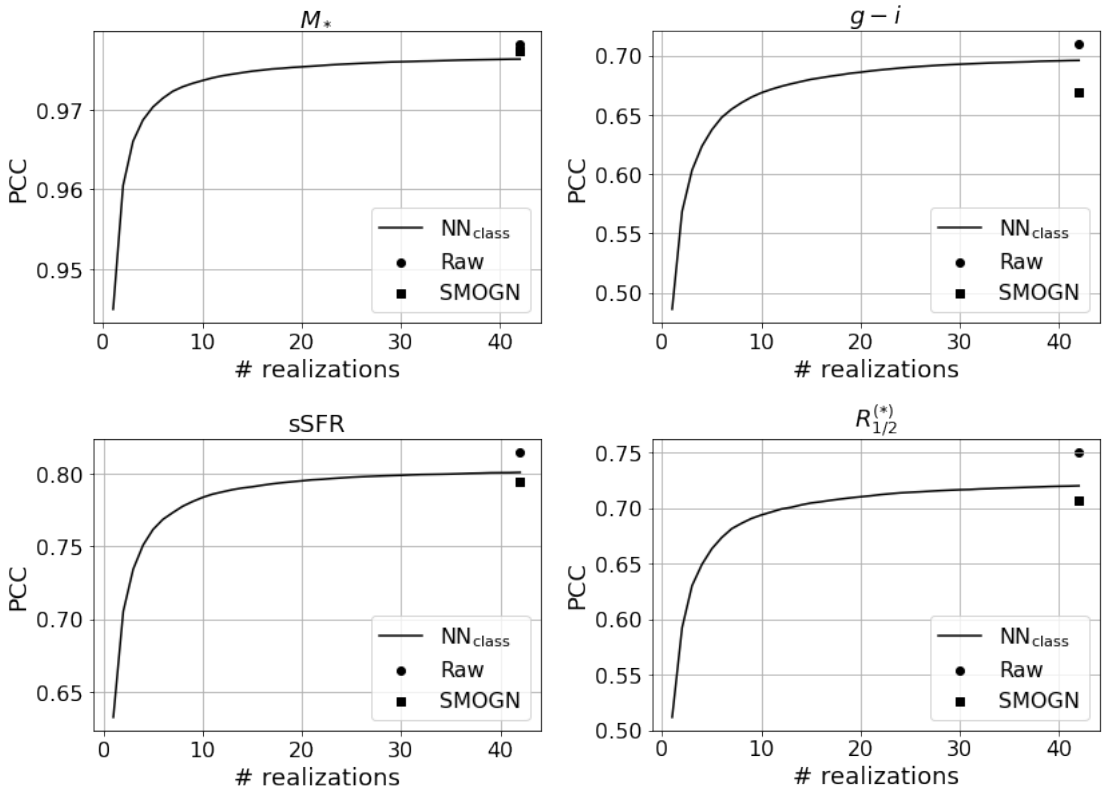

Figure 44 – **PCC comparison for galaxy properties predicted by Raw, SMOGN, and NNClass.** The PCC values of the baseline models Raw and SMOGN are shown as dotted and squared markers, respectively. *Source:* Reference (69).

5.4.3 Discussion and conclusions

Although there is an obvious relation between the baryonic and DM components of halos, there is also mounting evidence that the properties of halos alone are insufficient to reproduce the properties of galaxies, since the latter are shaped by a variety of galaxy-formation processes. ML regression models are traditionally designed to reproduce single-value statistics, and thus are ill-equipped to encode the intrinsic scatter in the halo-galaxy connection.

In order to alleviate the deficiencies of ML deterministic regression models, we have tested a different approach for the first time in the context of the halo-galaxy connection. The NNs are now trained to predict probability distributions instead of single-value statistics by means of a binning classification scheme. In essence, the distributions of galaxy properties are split into K narrow bins so that the NNs can associate a score to each of the K classes. This is performed in such a way that the output can be used as a proxy for the probability distributions of the central galaxy properties.

We have shown that this approach is in fact capable of producing bivariate distributions of galaxy properties, i.e., $P(Y_1, Y_2)$, in outstanding agreement with those from TNG300 (here, $\{Y_1, Y_2\}$ is any pair of galaxy properties). The predicted probability distributions yields significantly better results compared with the deterministic approach of our previous work

(see Subsection 5.3 and Reference (65)), as both a visual inspection and the 2D K-S test reveal. As a reference, our 2D K-S test for the joint distributions $P(Y_1, Y_2)$ yields performance results that are better by factors of $10 - 30$ as compared to those reported in Reference (65) and in Section 5.3.2.4. We have also checked that predicting jointly galaxy properties is particularly advantageous for the color–sSFR joint distribution.

Finally, the comparison of the presented method with the two previous approaches (Raw and SMOGN methods) using the single-value estimation shows that, after multiple realizations, we achieve similar scores. This result indicates that NNCLASS behaves as an effective maximum likelihood estimator. However, averaging over the sorted predictions does not accurately reflect the overall distributions, failing to represent the true distribution for the TNG300 data set and not yielding remarkable improvements in this regard. Nevertheless, this highlights the flexibility of using the results from the presented method over single-value estimators: one can select "single-values" from the predicted distributions by either randomly selecting a value from it or by averaging (or applying any other statistic) over the sorted distribution values.

6 FIELD-LEVEL LIKELIHOOD-FREE INFERENCE WITH GRAPH NEURAL NETWORKS

ΛCDM is the current standard model of Cosmology, describing well the evolution and expansion of the Universe (see Section 2.1). This model elucidates how primordial density perturbations in the early Universe were amplified by gravity, eventually leading to the formation of the large-scale structures observed today. To achieve this, the model relies on several cosmological parameters that characterize the composition and other fundamental properties of our Universe. One such parameter is Ω_m, which quantifies the fractional energy density of matter (both dark and baryonic matter). Obtaining accurate constraints for Ω_m is crucial for enhancing our understanding of the foundational physics governing the Universe.

Historically, statistics utilized to analyze the density and velocity fields of matter and galaxies have served as valuable probes for determining Ω_m (264, 304). This includes, e.g., the analysis of redshift-space distortions in galaxy redshift surveys, which arise from virial and peculiar velocities deviating from the homogeneous cosmic flow (305). These distortions significantly impact the statistical properties of galaxy clustering by breaking the symmetry between distances across and along the line-of-sight direction. As a result, these anisotropies directly probe the growth factor, which depends on Ω_m, as described in References (306, 307). In summary, the majority of these analyses consider the parameter constraints from large-scales. On the other hand, examples such as considering the pairwise velocity metric defined for galaxies and galaxy clusters as the peculiar velocity difference of pairs along their radial separation vector demonstrate that valuable cosmological information is embedded on the small scales (308, 309). However, there is a lack of techniques to precisely extract cosmological information from small scales.

Traditional methods have been widely used to estimate Ω_m. However, they often encounter challenges due to the necessity of a summary statistics (which we still do not know what of them contain all, or the majority, of the cosmological information (32–35)) and requirement for numerous realizations of computationally expensive simulations (36–40) (as discussed in Sections 2.3.5 and 2.3.5.1). Recently, ML techniques have demonstrated good performance compared to the traditional methods, sometimes even incorporating them (42). Specifically, *field-level likelihood-free inference*[1] methods operate by directly utilizing data from simulations (forward models), bypassing the need for summary statistics, and then inferring a posterior distribution over the parameters. Several studies have illustrated competitive results achieved by these ML techniques in comparison to conventional statistical inference methods (44–52).

[1] The term "field-level" is used here to refer to the fact that the ML suite takes as input the galaxy field, as opposed to feeding it with a summary statistic.

Specially attention has been directed towards GNN inferences (as discussed in Section 3.3.4) due to their ability to handle sparse and irregular data, without imposing restrictions on considered physical scales. Moreover, GNNs readily incorporate various physical symmetries, such as translational and rotational invariance, into their framework. For instance, researchers in Reference (50) demonstrated that GNNs achieved approximately $\sim 10\%$ accuracy in inferring Ω_m solely based on galaxy properties (e.g., positions, stellar mass, radius, and metallicity), without the need for summary statistics and through likelihood-free inference. However, their model exhibited a lack of robustness, which could be attributed to intrinsic disparities in subgrid models across different simulations or the models learning specific numerical artifacts. Conversely, when dealing with DM halos, authors in Reference (52) showed that positions and velocities remained robust to numerical variations in N-body codes and changes in astrophysical parameters when inferring Ω_m using a field-level approach.

In this chapter, we demonstrate the utilization of galaxy and halos phase-space information, reminiscent of historical concepts, to predict Ω_m by converting it into graphs and feeding GNNs. Our primary objective is to develop robust models. Section 6.1 outlines our efforts working with galaxy catalogs, while Section 6.2 focuses on training the machinery on halos and interpreting the GNN task into equations using SR. Furthermore, we delve into the reasons behind the success of Astrid data set in Section 6.1.6.3.1. Finally, we extend our efforts presented in Section 6.1 by incorporating real observational effects into the galaxy catalogs and assessing their impact in Section 6.3. In this regard, the work presented in this chapter is related to the background seen in Section 3.3.4. Additionally, the main achievements related to this work are associated with the publications of References (71, 74, 310, 311).

The recent impact of these efforts on the scientific community is evident in the utilization of GNNs to explore other astrophysical relations, such as the most important scales to halo-galaxy connection (312). Authors have identified a critical scale of approximately 3 Mpc, with important environmental information at 10 Mpc, for IllustrisTNG, of 205 Mpc$/h$. Another noteworthy development emerging as a competitive method subsequent to our findings involves the application of ML to predict parameter inference on simulated data catalogs, which incorporate the BOSS geometry and observational effects (313). In that work, the authors employ CNNs to compress galaxy information, considering survey systematics, and integrate it with Neural Posterior Estimation (NPE) to infer the cosmological parameters. Despite achieving impressive predictions, their method remains dependent on scales, as it requires discretizing the galaxy field to compress information within the CNNs.

6.1 Robust field-level likelihood-free inference with galaxies

In this section we showcase the achievements outlined in Reference (71). We train GNNs to conduct field-level likelihood-free inference using galaxy catalogs sourced from hydrodynamic simulations of the CAMELS project. The data utilized in this endeavor is elaborated

upon in Section 6.1.1, with data pre-processing detailed in Section 6.1.1.2, graph construction methods are explained in Section 6.1.1.3, and the architecture of the GNN expounded upon in Section 6.1.2. Our models exhibit the capability to infer the value of Ω_m with approximately 12% precision, demonstrating robustness to variations in astrophysics, subgrid physics, and subhalo/galaxy finder methodologies. These results will be presented and discussed in Sections 6.1.6 and 6.1.8. Additionally, we delve into an investigation of which galaxy properties remain robust and elucidate their contributions to the network predictions. Remarkably, we find that leveraging only the phase-space information of the galaxies yields the best results, as detailed in Section 6.1.6.3.

6.1.1 Data

In this section, we describe the data we use to train, validate, and test our models. We emphasize that all the galaxy properties considered in this work are direct from the simulations. In this way, we are not performing any changes in order to consider realistic effects, such as taking into account errors in the peculiar velocities. These considerations will be addressed in Section 6.3.

6.1.1.1 Simulations

Table 7 – **Characteristics of the hydrodynamical simulations.**

Model	Usage	Number of simulations used	Mean number of galaxies per catalog	Reference
Astrid	Train, validate & test	1000(LH) + 27(CV)	1114	(154)
SIMBA	Train, validate & test	1000(LH) + 27(CV)	1093	(155)
IllustrisTNG	Train, validate & test	1000(LH) + 27(CV) + 1024(SB)	737	(278)
IllustrisTNG300	Test	1(LH)	799	(281)
Magneticum	Test	50(LH) + 27(CV)	3655	(157)
SWIFT-EAGLE	Test	64(LH)	1255	(158)

The galaxy catalogs we use to train, validate, and test our models come from thousands of hydrodynamic simulations of the CAMELS project (see Section 2.5.2) (314, 315). The hydrodynamic simulations have been run with different codes that solve the hydrodynamic equations differently and implement different subgrid models: IllustrisTNG (278, 316), SIMBA (155), Astrid (154), Magneticum (157), and SWIFT-EAGLE (158, 317). All the simulations follow the evolution of 256^3 DM particles and are initialized with 256^3 fluid elements from $z = 127$ down to $z = 0$ in periodic boxes of $25\ h^{-1}$Mpc on a side. The catalogs used in this work correspond to $z = 0$. The fiducial values of the cosmological parameters are: $\Omega_m = 0.3$, $\Omega_b = 0.049$, $h = 0.6711$, $n_s = 0.9624$, $\sigma_8 = 0.8$, $w = -1$, $M_\nu = 0$ eV.

The CAMELS simulations can be classified into different sets and suites depending on how their parameters are arranged and which code was used to run them. We start by classifying the catalogs into different sets:

- **Latin Hypercube (LH).** The simulations in this category have their cosmological and astrophysical parameter variations arranged in a LH that spans: $\Omega_m \in [0.1, 0.5]$ and $\sigma_8 \in [0.6, 1.0]$, $A_{SN1} \in [0.25, 4.0]$, $A_{SN2} \in [0.5, 2.0]$, $A_{AGN1} \in [0.25, 4.0]$, and $A_{AGN2} \in [0.5, 2.0]$. A_{SN} and A_{AGN} are astrophysical parameters that control the efficiency of SN and AGN feedback (see References (311, 314) for a detailed description of the meaning of the astrophysical parameters in every simulation suite). Each of the simulations in the LH has been run with a different initial random seed for the generation of the initial conditions. We used these simulations for training, validating, and testing.

- **Cosmic Variance (CV).** These simulations have been run with the fiducial value of the cosmological and astrophysical parameters. The initial conditions for each simulation in this set have been generated with a different initial random seed. These simulations are only used for testing the models.

- **Sobol Sequence (SB).** The simulations in this set have their cosmological and astrophysical parameters arranged in a Sobol sequence (318). A total of 28 parameters are varied: 5 cosmological (Ω_m, Ω_b, h, n_s, σ_8) and 23 astrophysical. The astrophysical parameters varied include the usual ones ($A_{SN1}, A_{SN2}, A_{AGN1}, A_{AGN2}$) and incorporate many others such as star formation, galactic winds, BH growth, and quasar parameters. All of them vary in ranges around the fiducial values used in the IllustrisTNG set. Their range of variation is large enough to enable a broad sampling of the considered parameter (311). We note that this set covers the largest region in parameter space within CAMELS although at a much lower density given the high dimensionality of the considered space. We use these simulations only for testing and to investigate how well our models generalize.

The CAMELS simulations can also be classified into different model suites according to the code used to run them (different subgrid physical models–see Section 2.5.2):

- **IllustrisTNG.** These simulations were run using Arepo (319, 320) applying the same subgrid physics as the IllustrisTNG simulations (278, 316). This suite contains $1,000$ LH, 27 CV, and $1,024$ SB simulations[2].

- **SIMBA.** These simulations were run with the Gizmo code (321) and employ the same subgrid physics as the SIMBA simulation (155). This suite contains $1,000$ LH and 27 CV simulations.

[2] Note that, in the work of Section 6.3, we had access to the new SB28 suite which contains $2,048$ realizations for the SB set.

- **Astrid.** These simulations were run using MP-Gadget (322) applying some modifications to the subgrid model employed in the Astrid simulation (154, 311, 323). This suite contains $1,000$ LH and 27 CV simulations.

- **Magneticum.** These simulations were run with the parallel cosmological Tree-PM code P-Gadget3 (324), employing subgrid physical models according to the References (163, 325–334). The set contains 50 LH and 27 CV simulations.

- **SWIFT-EAGLE.** These simulations have been run with the Swift code (317, 335) using a new subgrid physics model based on the original Gadget-EAGLE simulations (158, 336), with some parameter changes (337). This suite contains 64 LH simulations.

Finally, to quantify the robustness of our model to super-sample covariance effects, we made use of the IllustrisTNG300-1 simulation (281), which covers a larger volume of $(205\ h^{-1}\text{Mpc})^3$ with slightly higher resolution than our fiducial CAMELS simulations and has a slightly different cosmology: $\Omega_m = 0.3089$, $\Omega_b = 0.0486$, $\Omega_\Lambda = 0.6911$, $h = 0.6774$, $\sigma_8 = 0.8159$, and $n_s = 0.9667$. This simulation was run with Arepo and made use of the IllustrisTNG subgrid physics model (278, 282–284, 338, 339).

We emphasize that although the name of the parameters $A_{SN1}, A_{SN2}, A_{AGN1}, A_{AGN2}$ is common among different simulations, their actual implementation and effect on galaxy properties and clustering can be very distinct. Therefore, it is important to keep in mind that those parameters are not meant to share physical effects, only their names.

6.1.1.2 Galaxy catalogs

Halos and subhalos are identified in the simulations for every snapshot using two different halo and subhalo finders: SubFind (134, 135) and VELOCIraptor (137, 138) (see Section 2.5.1.8). All galaxy catalogs are from SubFind with the exception of SWIFT-EAGLE, which only contains VELOCIraptor catalogs. The reason for using two different codes is to check the robustness of our results to the subhalo finding procedure, which can cause some differences in the number of galaxies and resolving substructures as shown in References (132, 133).

Galaxies are defined in all cases as subhalos that contain at least one star particle. In this work, we only consider galaxies with stellar masses above $1.3 \times 10^8\ M_\odot/h$. A galaxy catalog is constructed by taking all galaxies whose stellar mass is higher than a given threshold. For every simulation, we produce several galaxy catalogs by varying the stellar mass threshold.

A summary of the simulation characteristics can be found in Table 7, where we present their usage, the number of catalogs, the mean number of galaxies per catalog and the reference for each of the original galaxy formation models.

6.1.1.3 Galaxy graphs: construction

The input for our GNNs are the galaxy catalogs converted to graphs (see Section 3.3.4.1). We construct the graphs with the galaxy positions and their peculiar velocities (only the z component); in some models, we also include the stellar mass of the galaxies.

We follow the method presented in Reference (50) (and used in References (52) and (51) for halos) where galaxies represent the graph nodes and two galaxies are connected by an edge if their distance is smaller than a given linking radius r_{link}. As presented in Section 3.3.4.1, a similar approach could involve taking into account the k nearest neighbors for each node, to link them. However, for this particular application, it might overlook some neighbors in a clustered region that should be connected, while including others that are much farther away, and thus should have less influence on the node. We have verified that the performance slightly decreases when using this approach instead of considering nodes within a certain distance. Additionally, we use as a global property of the graph the logarithm of the number of galaxies in the graph: $\log_{10}(N_g)$[3].

We investigate the contribution of the z component of the galaxy's peculiar velocities v_z and the stellar mass $M_\star$ as node attributes. We transform these features according to:

$$v_z \to \mathrm{sign}(v_z) \cdot \log_{10}\left[1 + \mathrm{abs}(v_z)\right], \tag{6.1}$$

$$M_\star \to \log_{10}(1 + M_\star)\ . \tag{6.2}$$

We chose to work with only one component for the galaxy velocity. This is because we want to be as close as possible to observational data, where we have access only to the radial peculiar velocity, i.e., the velocity measured along the line of sight.

The edge features contain information about the spatial distribution of galaxies (their positions), and those properties are designed to make the graph invariant under rotations and translations. We follow Reference (50) and set the edge features as:

$$\mathbf{e}_{ij} = \left[\frac{|\mathbf{d}_{ij}|}{r_{link}}, \alpha_{ij}, \beta_{ij}\right] = \left[\gamma_{ij}, \alpha_{ij}, \beta_{ij}\right], \tag{6.3}$$

where

$$\mathbf{d}_{ij} = \left[\mathbf{r}_i - \mathbf{r}_j\right], \tag{6.4}$$

$$\boldsymbol{\delta}_i = \mathbf{r}_i - \mathbf{c}, \tag{6.5}$$

$$\alpha_{ij} = \frac{\boldsymbol{\delta}_i}{|\boldsymbol{\delta}_i|} \cdot \frac{\boldsymbol{\delta}_j}{|\boldsymbol{\delta}_j|}, \tag{6.6}$$

$$\beta_{ij} = \frac{\boldsymbol{\delta}_i}{|\boldsymbol{\delta}_i|} \cdot \frac{\mathbf{d}_{ij}}{|\mathbf{d}_{ij}|}\ , \tag{6.7}$$

$$\gamma_{ij} = \frac{|\mathbf{d}_{ij}|}{r_{link}}\ , \tag{6.8}$$

[3] We have checked that including the number of galaxies as global feature yields slightly better results. For that reason, we keep that property.

with $\mathbf{r}_i$ representing the position of a galaxy i and $\mathbf{c} = \sum_i^N \mathbf{r}_i/N$ being the centroid. Here, the *distance* $\mathbf{d}_{ij}$ is the difference of two galaxy (i and j) positions, the *difference vector* $\boldsymbol{\delta}_i$ denotes the position of a galaxy i with respect to the centroid, α_{ij} is the (cosine of) the angle between the difference vectors of two galaxies, while β_{ij} represents the angle between the difference vector of a galaxy i and its distance to another galaxy j. We account for PBC when computing both distances and angles. Moreover, we consider reverse edges and we do not consider self-loops. Note that, by construction, the model is rotational and translation invariant, as those operations will not change the edge features of the graph. In other words, they will remain the same while performing the usual rotation and translational matrix transformations to the galaxy positions (50).

In Figure 45 we show graphs constructed from galaxy catalogs of the different simulations: Astrid, SIMBA, IllustrisTNG, SB28, Magneticum, and SWIFT-EAGLE. All these catalogs contain galaxies with minimum stellar mass: $M_\star = 1.95 \times 10^8\ M_\odot/h$. In all the graphs galaxies are colored according to their v_z (transformed according to Equation 6.1), and two galaxies are connected by a black line if their distance is within $r_{\rm link} \simeq 1.25\ h^{-1}$Mpc (this value was found with Optuna, as it will be described in Section 6.1.4). Notice that we are not connecting galaxies which are linked due to the PBC in this representation, i.e., a galaxy near the border of the box is not showing to be connected to some other galaxy in the other box extreme, even when they are linked due to these conditions. This simple visual comparison shows that the spatial distribution of galaxies and their peculiar velocities are similar among all simulations. We note that the graph constructed from the Magneticum simulation exhibits a significantly larger number of galaxies than the others; this happens due to the employed AGN model used in Magneticum.

Every graph is characterized by a set of labels that we aim at inferring (Ω_m). We normalize these labels as θ_i, using

$$\theta_i \to \frac{(\theta_i - \theta_{min})}{(\theta_{max} - \theta_{min})}, \tag{6.9}$$

where θ_{min} and θ_{max} represent the minimum and the maximum values of the corresponding parameter.

In Figure 46 we compare the number of galaxies in the LH catalogs for the different CAMELS simulations, considering a threshold in stellar mass as $M_\star = 1.95 \cdot 10^8\ M_\odot/h$. In almost all the cases the mean number of galaxies is $\sim 1,000$, being a bit lower (~ 700) for IllustrisTNG and its variation SB28, and dramatically higher ($\sim 3,600$) for Magneticum. Also, we can see that Astrid includes catalogs with a huge range of galaxy number ($N \in [30, 5,000]$), while the SIMBA and IllustrisTNG LH sets are much narrower (the same follows for SB28, with a higher dispersion of galaxy number, but not so broad as in Astrid). Finally, the range of the number of galaxies for Magneticum is $N \in [1,000; 5,500]$, including catalogs with such a large number of galaxies that do not have equivalent simulations in the SIMBA and

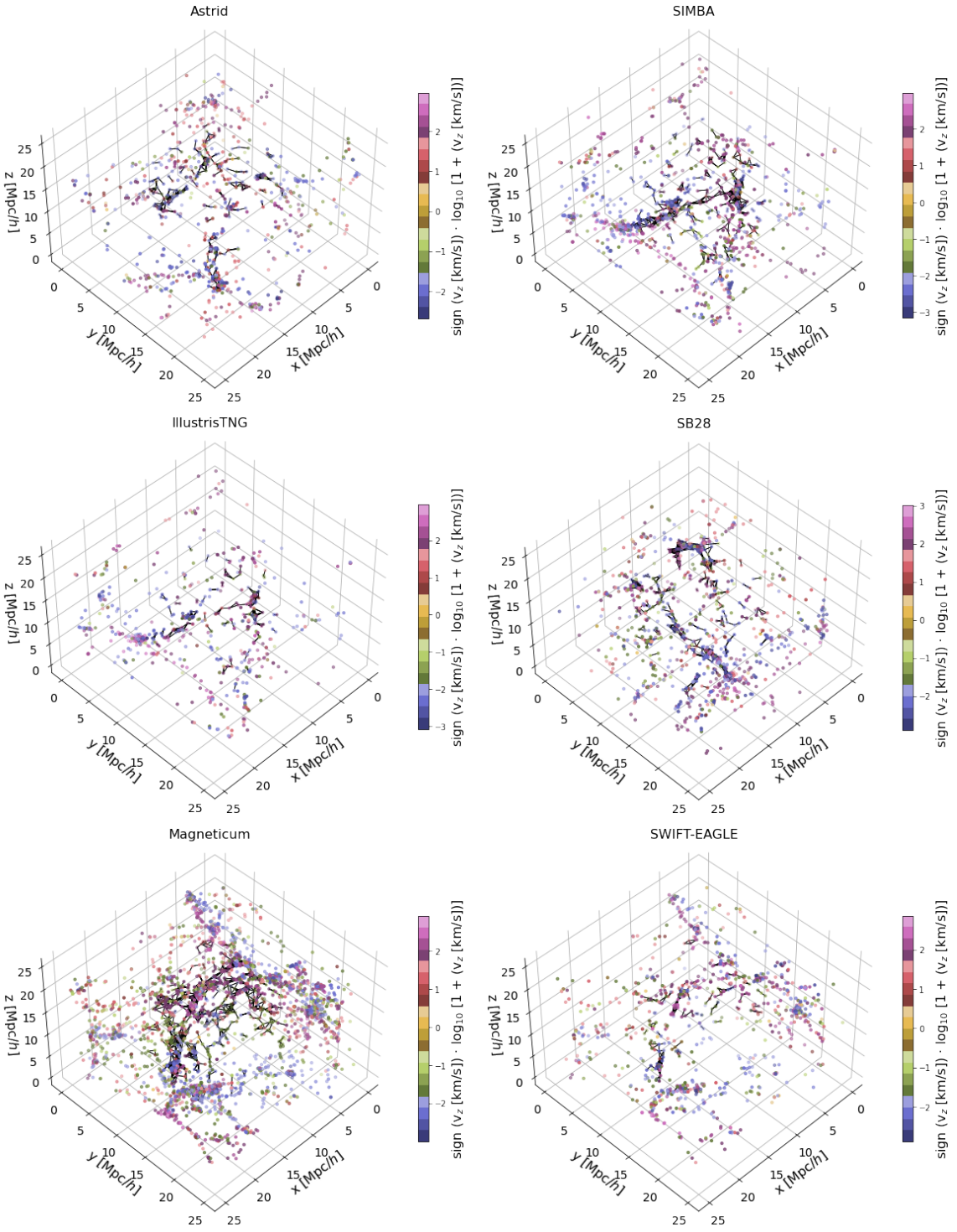

Figure 45 – **Examples of graphs constructed from galaxy catalogs from different CAMELS simulations.** We present graphs of Astrid, SIMBA, IllustrisTNG, SB28, Magneticum, and SWIFT-EAGLE. The nodes represent the galaxies and their colors correspond to the normalization (Equation 6.1) of the z component of their peculiar velocity. Galaxies are connected by edges (shown as black lines) if their distance is smaller than $r_{\rm link} \sim 1.25\ h^{-1}$Mpc. We stress that we are not connecting the galaxies which are linked due to PBC in these plots. *Source:* Reference (71).

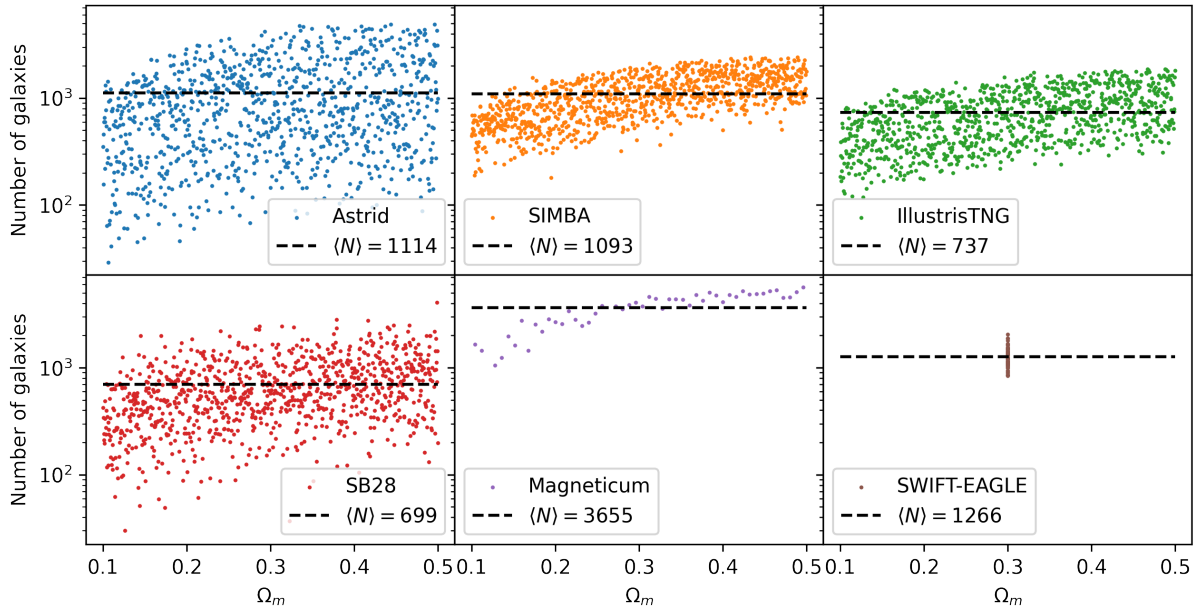

Figure 46 – **Comparison of the number of galaxies per LH catalog in CAMELS simulations.** We present the results for Astrid (top left), SIMBA (top middle), IllustrisTNG (top right), SB28 (bottom left), Magneticum (bottom middle), and SWIFT-EAGLE (bottom right). The horizontal lines correspond to the mean number of galaxies per simulation. *Source:* Reference (71).

IllustrisTNG data sets. As mentioned in Section 6.1.1.3 the large number of galaxies in Magneticum is related to the particular feedback model employed in those simulations.

The distances and the number of edges among the galaxies belonging to different catalogs were also investigated, as well the percentage of single galaxies per catalog. As expected, the distances among galaxies cover a range $d \in [10^{-2}, 21.65]\ h^{-1}$ Mpc. All the catalogs have a similar shape in their spatial distributions, with small differences on small scales. Single galaxies (the ones which are not connected to any other, and therefore only contribute to the propagation of their node information), on average, do not correspond to more than $\sim 20\%$ of the galaxies in the catalogs. This means that most of the information of the galaxies came from their connections (i.e. clustering properties). The number of edges per catalog is of order $\sim 10,000$, indicating that most galaxies have ~ 10 connections. Finally, the r_{link} found in all the models, for all different CAMELS sets in the hyperparameter training optimization, was around $1.25\ h^{-1}Mpc$.

6.1.2 GNN architecture

The architecture we employ in this work follows the one presented in CosmoGraphNet[4] (50). We have used a *message passing scheme* where each message passing layer updates the node and edge features (see MetaLayer block and Equations 3.11, 3.12, and 3.13 in Section

[4] Available on Github repository https://github.com/PabloVD/CosmoGraphNet, DOI: 10.5281/zenodo.6485804.

3.3.4.2.1). The number of layers to perform this update is a hyperparameter to be chosen in the optimization scheme. We also made use of *residual* in the intermediate layers. The use of residuals means adding the input of the layer to its respective output, i.e., adding node/edge attributes to node/edge models. A discussion about this use can be found in References (340) and (50).

Once the graph has been updated using the N message passing layers, we collapse it into a 1-dimensional feature vector using

$$\mathbf{y} = \mathcal{F}\left(\left[\bigoplus_{i\in\mathfrak{F}} \mathbf{n}_i^N, \mathbf{g}\right]\right), \tag{6.10}$$

where $\mathcal{F}$ is the last MLP, $\oplus_{i\in\mathfrak{F}}$ the last multi-pooling operation (done exactly according to Equation 3.13, but operating over all nodes in the graph $\mathfrak{F}$), and $\mathbf{y}$ represents the target of the GNN (e.g. Ω_m).

All the MLP are constructed by a series of fully connected layers with ReLU activation function (except for the last layer, which does not employ an activation function). The number of layers, the number of neurons per layer, the weight decay, and the learning rate were considered as hyperparameters. The implementation of all the architectures presented in this work was done using PyTorch Geometric (222).

6.1.2.1 Variations of the architecture

In Section 6.1.6.3 we investigate whether the information of our model is due to clustering, the distribution of velocities, or both. For that test, we made use of slightly different architectures to the one outlined above. Their main differences are:

- **Galaxy positions.** This model is used to quantify how much information is coming from the clustering of galaxies, i.e., it only uses galaxy positions. For that reason, the graphs only contain edge features (in the same way outlined above) and no node features. Because of this, the first layer of the model operates in a slightly different way, updating the edge and node models according to Equations 3.20 and 3.21, respectively (see Section 3.3.4.3.2). Note that other layers operate in exactly the same way as described in Equations 3.11-3.12.

- **Galaxy velocities.** This model is used to quantify how much information is coming from the distribution of galaxy velocities. Therefore, the graphs do not contain any spatial information and we can use *deep sets* (227) architecture (see Section 3.3.4.3.1). In this case, we only have a node model according to Equation 3.19.

No matter the variation in the architecture, the target quantity is computed using Equation 6.10.

6.1.3 Likelihood-free inference and the loss function

Our models are a mix of a GNN together with a MNN (see Section 3.4.1). So, they are trained to infer the value of a given parameter (Ω_m) by predicting the marginal posterior mean μ_i and standard deviation σ_i without making any assumption about the form of the posterior. We do this following a modified loss function according to the Equation 3.25 (232).

We note that we will be referring to the error of the model as the quantity described above σ_i. This error only represents the *aleatoric error*, and therefore does not include the epistemic one, i.e., the error intrinsically related to the ML model (see Section 3.1). We have quantified the magnitude of the epistemic errors by training 10 different models with the same value of the hyperparameters (the best ones for the considered setup) and calculating the variance between the predictions of the models. We find that error to be $10\times$ smaller than the aleatoric one. Therefore, from now on, we will only report aleatoric errors since they dominate the total error budget.

6.1.4 Training procedure and optimization

We train our models on graphs constructed from galaxy catalogs of the LH sets of a given suite (e.g. the LH set of the Astrid simulations). We initially split the $1,000$ LH simulations into training (850 simulations), validation (100 simulations), and testing (50 simulations). For each simulation, we generate 10 galaxy catalogs constructed by taking all galaxies with stellar masses larger than $1.3R \times 10^8\ M_\odot/h$, where R is a random number uniformly distributed between 1 and 2. This strategy is made in order to marginalize over different minimum threshold values for stellar masses, as well to increase the number of catalogs used to train the models[5]. For each catalog, we produce a graph as outlined in Section 6.1.1.3.

We then train the models utilizing the above architecture for 300 epochs making use of Adam optimizer (203) to perform the gradient descent, and a batch size of 25 samples. The hyperparameter optimization (where we have used the learning rate, the weight decay, the linking radius, the number of message passing layers, and the number of hidden channels per layer of the MLPs) was carried out using the Optuna package (243) to perform a Bayesian optimization with TPE (244) (see Section 3.5.4). We made use of at least 100 trials to perform this task and we directed Optuna to minimize the validation loss, computed using an early-stopping scheme, in order to save only the model with the minimum validation error. The selected model was used for test subsequently.

[5] A similar trick was used in Reference (52), where the authors employed an augmentation in the halo catalogs, choosing them according to a minimum number of DM particles as a threshold. Authors from Reference (310) also made use of this method.

6.1.5 Performance Metrics

We quantify the accuracy and precision of our models using different metrics that we describe below. We consider the true value of the parameter in question for graph i as θ_i, while we denote as μ_i and σ_i the prediction of the network for the posterior mean and standard deviation, respectively.

- **Root Mean Squared Error (RMSE):**

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\theta_i - \mu_i\right)^2}. \tag{6.11}$$

  Low values of the RMSE indicate the model is precise.

- **Coefficient of determination:**

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(\theta_i - \mu_i\right)^2}{\sum_{i=1}^{N}\left(\theta_i - \bar{\theta}_i\right)^2}, \tag{6.12}$$

  where $\bar{\theta}_i = \frac{1}{N}\sum_{i=1}^{N}\theta_i$. Values close to 1 indicate the model is accurate.

- **Pearson Correlation Coefficient (PCC):**

$$\text{PCC} = \frac{\text{cov}\left(\theta, \mu\right)}{\sigma_\theta \sigma_\mu}. \tag{6.13}$$

  This is the same PCC as the one presented in Equation 5.3, but in terms of this work.

- **Bias:**

$$b = \frac{1}{N}\sum_{i=1}^{N}\left(\theta_i - \mu_i\right). \tag{6.14}$$

  This statistic quantifies how much the inferences are "biased" with respect to the truth values; better values are close to 0.

- **Mean relative error:**

$$\epsilon = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\theta_i - \mu_i\right|}{\mu_i}. \tag{6.15}$$

  Low values of this statistic indicate the model is precise.

- **Reduced chi squared:**

$$\chi^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\theta_i - \mu_i}{\sigma_i}\right)^2. \tag{6.16}$$

  This statistic quantifies the accuracy of the estimated errors. Values of χ^2 close to 1 indicate the magnitude of the errors (posterior standard deviation in our case) is properly inferred, while values larger/smaller than 1 indicate the model is under/over predicting the errors.

We make use of these statistics to quantify the accuracy, precision, and bias of a given model in the test set. Note that in some cases we omit to report the value of some of these statistics for clarity, or when the statistics are not well defined (e.g. when tested on the CV set).

6.1.6 Results

In this section we present the main results of testing our GNN models on galaxy catalogs with different cosmologies, astrophysical parameters, and subgrid physic models from the catalogs used for training. We start by showing the results of our best model, which only needs 3D galaxy positions and 1D velocity components, in Section 6.1.6.1. We then attempt to increase the precision of the model by adding more galaxy properties, particularly stellar mass, in Section 6.1.6.2. Next, we investigate the origin of the information extracted by our models in Section 6.1.6.3.

Note that we focus our analysis entirely on Ω_m. This is because our constraints on σ_8 are very weak. We provide further details on this in Section 6.1.7. All results below are shown for catalogs built with galaxies with a minimum value of stellar mass as $M_\star = 1.95 \cdot 10^8\ M_\odot/h$, a value right in the middle of the threshold used in our training criteria[6].

6.1.6.1 Positions & velocities

We start by showing the results of training GNNs on catalogs that only contain the positions and velocities (solely the z component)[7] of galaxies to infer the value of Ω_m. We have trained models using galaxy catalogs from the LH sets of the Astrid (and present the results in Section 6.1.6.1.1), IllustrisTNG, and SIMBA (with results presented in Section 6.1.6.1.2) simulations. We then test these models on all other galaxy catalogs not included in their training set.

6.1.6.1.1 Astrid results

We found that the model trained on Astrid galaxy catalogs exhibits the best extrapolation properties. The success of the model trained on Astrid can be associated with:

1. The variety in the number of galaxies along the Astrid catalogs in LH sets, which vary from small to large numbers of galaxies ($N \in [30; 5,000]$–see more details in the Section 6.1.1.3 and Figure 46).

2. Astrid produces larger variations in some galaxy properties given the parameter variations in the LH set (see Section 6.1.6.3.1) (311).

[6] We have checked that our results are not very sensitive to the particular stellar mass cut we take, as long as we are not very close to the training boundaries.

[7] Due to homogeneity and isotropy, the results presented choosing the z component of the velocity are equivalent to choosing either x or y ones.

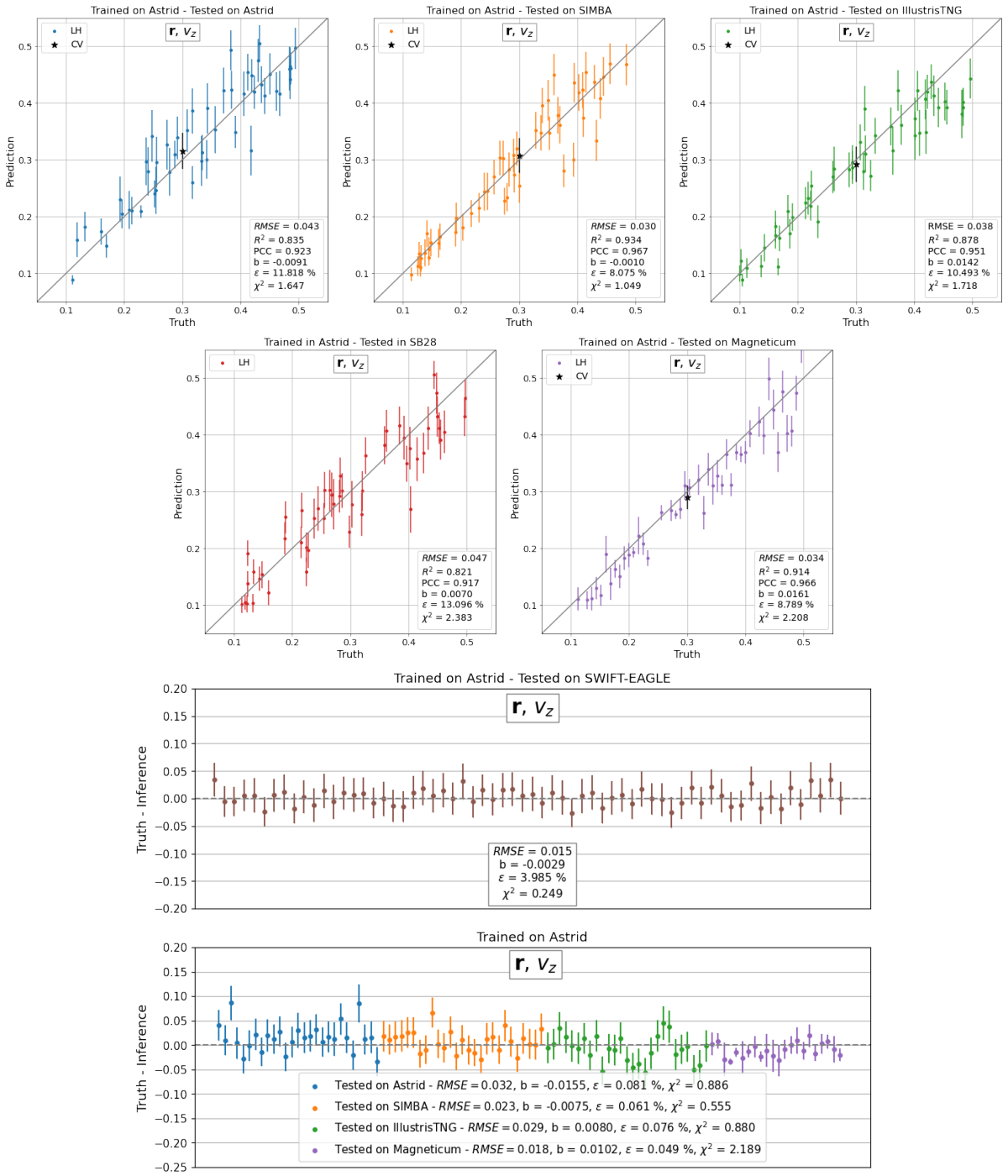

Figure 47 – **Ω_m predictions for a model trained on Astrid, using galaxy positions and velocities in the z direction.** We present the results for a model *trained* on Astrid and *tested* on Astrid (top left), SIMBA (top middle), IllustrisTNG (top right), SB28 (second row left), Magneticum (second row right), and SWIFT-EAGLE (third row). The bottom panel shows the results of testing on CV sets of Astrid, SIMBA, IllustrisTNG, and Magneticum. *Source:* Reference (71).

In addition, we trained a model on SB28 set (to check if the wide range in number of galaxies, presented on SB28 catalogs as well, was the main reason of the success of the model–see Figure 46). Although, the model does not show good predictions when tested on the other simulations.

In Figure 47 we show the results of testing the model on galaxy catalogs from the LH sets of Astrid (top left), SIMBA (top middle), IllustrisTNG (top right), SB28 (second row left), Magneticum (second row right), and SWIFT-EAGLE (third row). In all these plots (apart from SB28 and SWIFT-EAGLE) we present the average (of their mean and standard deviation) across all of their CV boxes as a black point at $\Omega_m = 0.3$. The results of testing the model on galaxies catalogs from the CV set of the different suites are shown in the bottom panel. Note that for clarity we only show 50 randomly selected samples of the predictions for all the LH results[8]. We stress that even if we only show the results for 50 random catalogs, the numbers reported for the different performance metrics (e.g. RMSE) are evaluated using all catalogs in the test set (e.g. $1,000$ catalogs for IllustrisTNG).

When using the model trained on Astrid and testing it on itself, we find that the GNN is able to infer Ω_m with $RMSE = 0.043$, $R^2 = 0.835$, $PCC = 0.923$, $b = -0.0091$, $\epsilon = 11.8\%$, and $\chi^2 = 1.647$. These numbers indicate the model is accurate, precise, unbiased, and its errors are only slightly under predicted[9]. While testing that model on the other simulations the performance metrics are in the ranges: $RMSE \in [0.015, 0.047]$, $R^2 \in [0.821, 0.934]$, $PCC \in [0.917, 0.967]$, $b \in [-0.0010, 0.0161]$, $\epsilon \in [4.0, 13.1]\%$, and $\chi^2 \in [0.249, 2.383]$, showing that the model extrapolates very well, as can also be seen in Figure 47. Note that the model performs best on SIMBA and SWIFT-EAGLE, and worst on SB28. This indicates that, while the model is generally robust, even when tested on SB28, it becomes increasingly difficult to extrapolate predictions over distant regions in parameter space.

We have included a test using IllustrisTNG300 box in order to estimate the importance of super-sample covariance effects. Basically, the lack of power on scales larger than our boxes can affect both the abundance and clustering of galaxies (90, 341–343). We find that our method can partially account for these effects. We provide further details in Section 6.1.6.1.3.

We now discuss the performance of the model on galaxy catalogs from the CV set. We find that our model works better when tested on the CV catalogs compared to the LH and SB sets. This could be due to the fact that the cosmology and astrophysics of those models lie exactly in the center of the training set. Those configurations are less prone to biased results, although it is interesting to observe that cosmic variance effects are not the main contribution to the error budget. Finally, all the different simulations end up with differences lower than 5%

[8] In the case of Astrid we only have 50 samples in the test set since the majority of the LH set was used for training.

[9] To show the scores for the best model while testing it on Astrid and Magneticum, we removed respectively 1 and 4 predictions that correspond to a χ^2 larger than 14.0. They are points in the test set that achieved this bad inference and that we call "outliers". Outliers not only because of the bad scores but mainly because they correspond to particular realizations in the LH set with extreme values for the astrophysical parameters, which are realizations far away from the fiducial model. We do not follow this procedure in the other models (apart from the best model, trained on Astrid using only galaxy positions and z component of the velocity) because they end up with a huge number of "bad" predictions, not only in the matter of fact to this issue.

(apart from some boxes of Astrid or SIMBA, where we achieve differences {truth - inference} up to 10%) for the best model, once again being accurate, precise, and without bias.

6.1.6.1.2 SIMBA and IllustrisTNG results

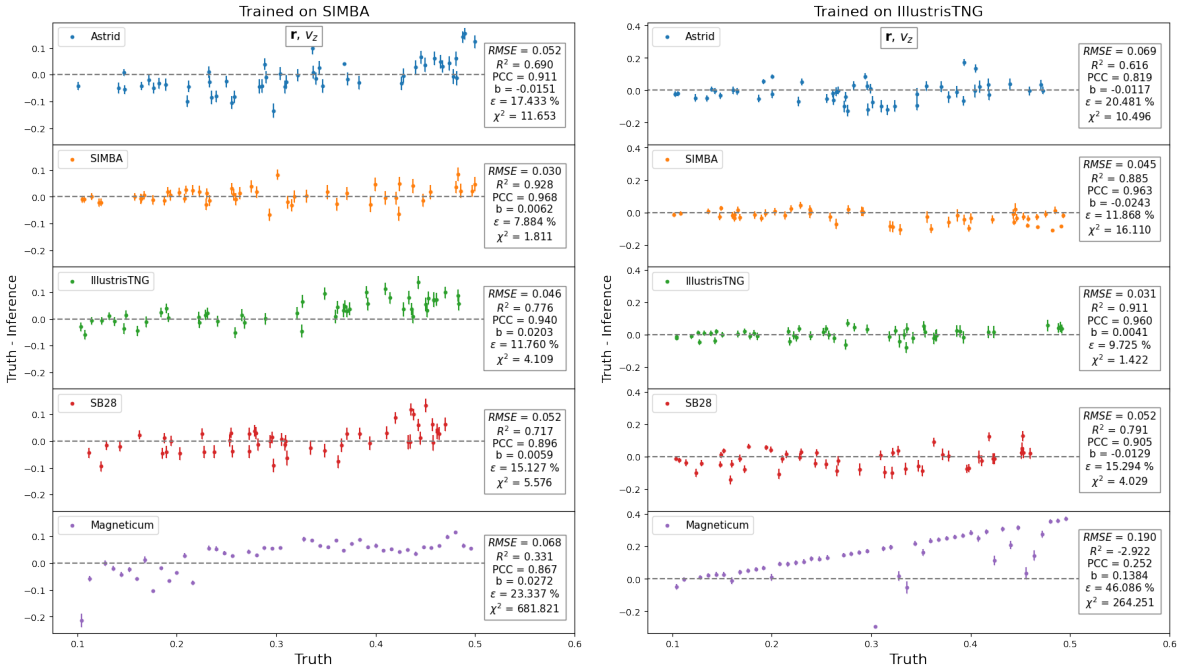

Figure 48 – **Ω_m predictions for a model trained on SIMBA and IllustrisTNG, using galaxy positions and velocities in the z direction.** We present the results for LH set tests of a model *trained* on SIMBA (on the left) and IllustrisTNG (on the right) and *tested* on Astrid, SIMBA, IllustrisTNG, SB28, and Magneticum respectively from the top to the bottom. *Source:* Reference (71).

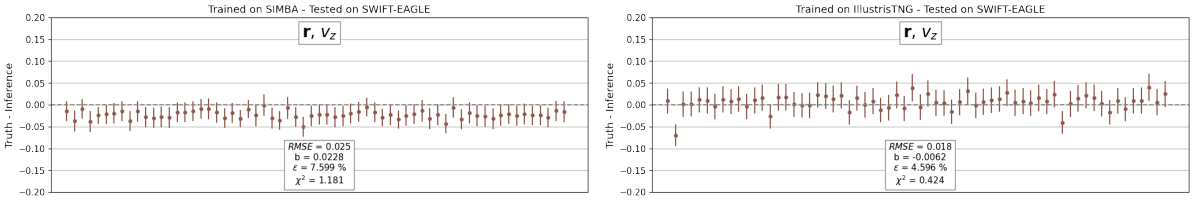

Figure 49 – **Ω_m predictions on SWIFT-EAGLE for a model trained on SIMBA and IllustrisTNG, on LH set, using galaxy positions and velocities in the z direction.** We present the results for a model *trained* on SIMBA (on the left) and IllustrisTNG (on the right) and *tested* on SWIFT-EAGLE. *Source:* Reference (71).

Here we present similar results to the ones presented in Section 6.1.6.1, for models trained using SIMBA and IllustrisTNG data sets. We stress that the GNN architecture follows the same structure as the one used in the best model (but with a different set of hyperparameters, also found using Optuna).

All the results are presented in Figures 48, 49 and 50, where we plot the values for {truth - inference} in the y-axis, while the x-axis shows either the truth values of Ω_m or an

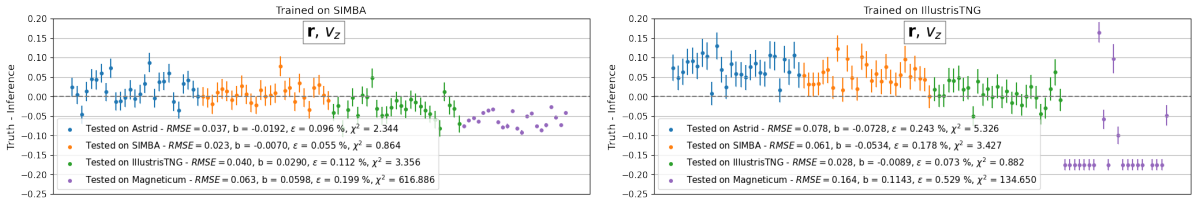

Figure 50 – **Ω_m predictions for a model trained on SIMBA and IllustrisTNG, on CV set, using galaxy positions and velocities in the z direction.** We present the results for CV set tests of a model *trained* on SIMBA (on the left) and on IllustrisTNG (on the right) and *tested* on Astrid, SIMBA, IllustrisTNG, SB28, and Magneticum. *Source:* Reference (71).

arbitrary order of the predictions by simulation suite. The metrics for the models trained on SIMBA/IllustrisTNG and tested on themselves are very good (even compared to the best model): $RMSE = [0.030, 0.031]$, $R^2 = [0.911, 0.928]$, $PCC = [0.960, 0.968]$, $b = [0.0041, 0.0062]$, $\epsilon = [7.9, 9.7]\%$, and $\chi^2 = [1.422, 1.811]$. However, all the tests on the other simulations are worse: $RMSE \in [0.018, 0.190]$, $R^2 \in [-2.922, 0.885]$, $PCC \in [0.252, 0.963]$, $b \in [0.0059, 0.1384]$, $\epsilon \in [4.6, 46.1]\%$, and $\chi^2 \in [0.424, 681.821]$. The worst predictions show up when the networks are tested on Magneticum (both, for the model trained on SIMBA and IllustrisTNG, but being worse for the latter). The tests on SWIFT-EAGLE and in the CV sets show that the scores are, in most cases, a bit worse compared to the best model (when we train the model using Astrid).

Our results suggest that the very poor predictions for Magneticum are due to the fact that the models trained on SIMBA and IllustrisTNG have never seen catalogs with such a high number of galaxies, which is the case for Magneticum catalogs (see Section 6.1.1.3, specially Figure 46, which shows that Astrid covers a large range of number of galaxies when compared to SIMBA and IllustrisTNG). We have tested to increase the stellar mass cut in Magneticum catalogs and have obtained better predictions (comparable to the same models tested on the other catalogs apart themselves) while using the models trained on SIMBA/IllustrisTNG. This shows that reducing the number of galaxies in Magneticum catalogs improves their inferences significantly. Therefore, although the number of galaxies is not the most important property in the analysis, we can clearly see their effect on the model predictions while taking a look at these results.

Finally, in contrast to the robust model that was trained using Astrid, the inferences from the models trained using SIMBA and IllustrisTNG are, unfortunately, not robust across different simulations.

6.1.6.1.3 Super-sample covariance analysis

We start noticing that our $25\ h^{-1}$Mpc boxes have a mean overdensity, $\langle \rho/\bar{\rho} \rangle = 1$. In the real Universe, $(25\ h^{-1}\text{Mpc})^3$ patches will not satisfy that equality, and values larger or smaller

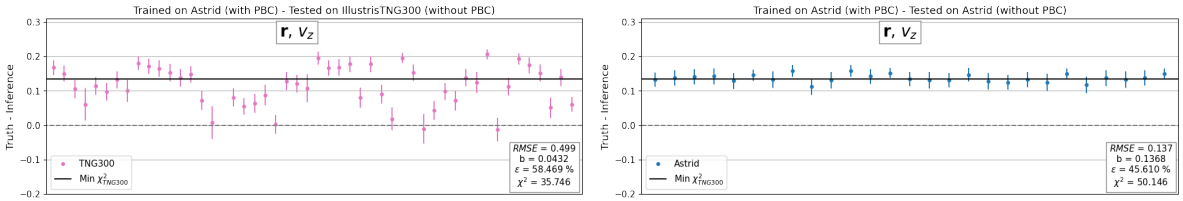

Figure 51 – **Ω_m predictions for a model trained on Astrid (with PBC), using galaxy positions and velocities in the z direction and tested considering PBC.** We present the results for a model *trained* on Astrid and *tested* on: (1) 50 random $(25\ h^{-1}\text{Mpc})^3$ sub-volumes of the IllustrisTNG300 simulation (on the left) and (2) Astrid (on the right). In both cases the model was trained considering the PBC and tested without this consideration. *Source:* Reference (71).

will appear due to the presence of power on modes larger than the size of that region. Those modes are expected to affect both the clustering of galaxies and their internal properties. Here we investigate whether such effects will affect our models. To test this, we made use of the IllustrisTNG300-1 simulation, which covers a periodic volume of $(205\ h^{-1}\text{Mpc})^3$ at a slightly higher resolution than the CAMELS simulations.

We have selected 50 random $(25\ h^{-1}\text{Mpc})^3$ sub-volumes within the IllustrisTNG300 box, taking the galaxies in those sub-volumes and constructed graphs to input into our model. It is important to note that we have turned off the PBC when constructing the graphs, due to the fact that the distribution of galaxies is not periodic within the sub-volumes. The results of testing our model with these galaxy catalogs are shown in Figure 51. We can see that the inferences for the IllustrisTNG300 catalogs have a positive bias of $b = 0.0432$ and the different estimations fluctuate around an offset that we indicate as Min χ^2_{TNG300}. This value represents the χ^2 minimization considering the IllustrisTNG300 inferences.

In order to check if this offset can be an effect of turning off the PBC we have tested the model on Astrid galaxy catalogs whose graphs have been constructed neglecting PBC. The results are presented in the right panel of Figure 51. We can see that we find almost the same offset for these new predictions.

Given the large effect that the PBC have on our results, we have retrained the GNN model on Astrid galaxy catalogs whose graphs are constructed without using PBC. We then test that model on galaxy catalogs from 100 random sub-volumes of the IllustrisTNG300 simulation. The results are presented in Figure 52. We can see that the inferences do not exhibit good scores: $RMSE = 0.089$, $b = -0.0073$, $\epsilon = 24.8\%$, and $\chi^2 = 34$. Even though, all the predictions fluctuate around the true values, indicating that we may have outliers. After removing predictions related to $\chi^2 > 14.0$ (35 points) we achieve better results that follows for: $RMSE = 0.059$, $b = -0.0118$, $\epsilon = 16.6\%$, and $\chi^2 = 4,0$.

From these results, we conclude that our method is not severely affected by super-sample covariance in the majority of the cases, although it does not work in all scenarios. We

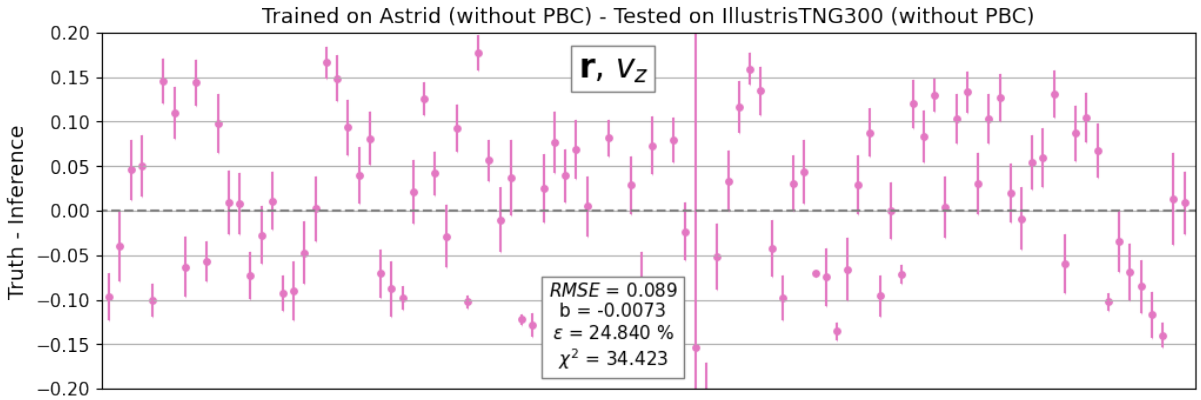

Figure 52 – **Ω_m predictions for a model trained on Astrid (without PBC), using galaxy positions and velocities in the z direction and tested removing PBC.** We present the results for a model *trained* on Astrid and *tested* on 100 random $(25\ h^{-1}\text{Mpc})^3$ sub-volumes within IllustrisTNG300. This specific model was trained without the PBC and tested without this too. *Source:* Reference (71).

note that the fraction of outliers (i.e. cases where the model performs badly) is much higher in this test case than in, e.g., SB28 simulations.

6.1.6.2 Positions, velocities, and stellar mass

We now investigate whether we can make our model more precise, while keeping it robust, by considering an additional galaxy property: *stellar mass*. For this, we construct graphs in the standard way (as described in Section 3.3.4.1) but taking as node features both velocity and stellar mass: $[v_z, M_\star]$ (properties normalized as described in Section 6.1.1.3). We then train GNN models using catalogs from the Astrid LH set.

We present the results in Figure 53. When testing the model on galaxy catalogs from the Astrid LH set we find that the results improved for almost all the metrics: $RMSE = 0.039$, $R^2 = 0.863$, $PCC = 0.936$, $b = -0.0090$, $\epsilon = 9.62\%$, and $\chi^2 = 1.849$, which means that the GNN was able to extract more information from the catalogs. On the other hand, when testing the model on the galaxy catalogs from the other simulation suites the scores worsen: $RMSE \in [0.032, 0.077]$, $R^2 \in [0.238, 0.926]$, $PCC \in [0.902, 0.966]$, $b \in [0.0096, 0.0651]$, $\epsilon \in [10.7, 20.7]\%$, and $\chi^2 \in [2.825, 14.167]$. In other words, the model has become more precise when tested on itself, at the expense of becoming less accurate, when tested on other simulation sets. It is worth noting that some metrics actually improved when tested on galaxy catalogs from Magneticum, as seen in Figure 53. It is not clear to us what could be the explanation behind this: whether it is either a coincidence or due to the fact that galaxies in Astrid and Magneticum are more alike somehow while considering this specific galaxy property.

Additionally, our results are in agreement with those of Reference (50) who performed a similar analysis with galaxy catalogs whose node features were the maximum circular velocity,

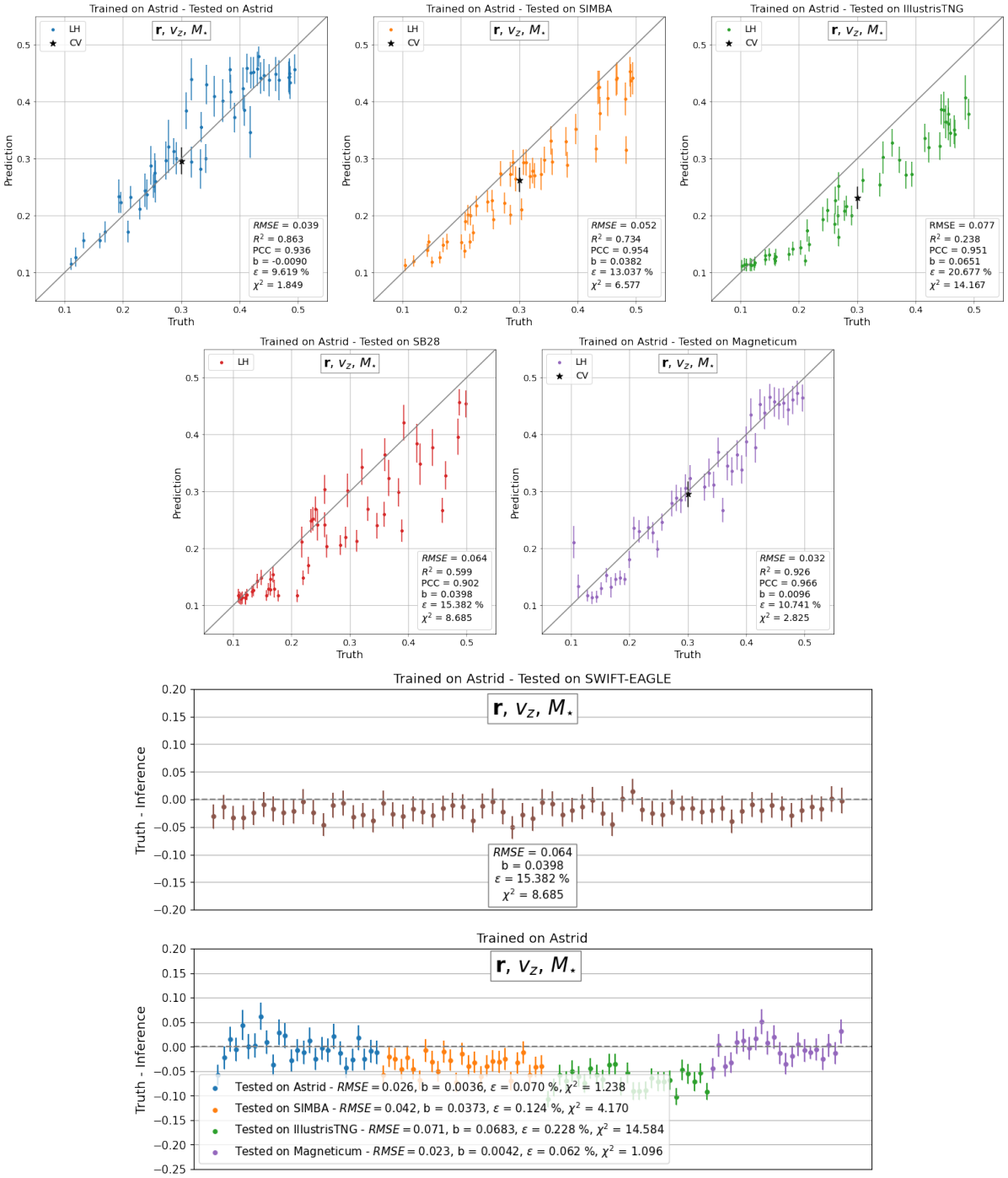

Figure 53 – **Ω_m predictions for a model trained on Astrid, using galaxy positions, velocities in the z direction, and stellar mass.** We present the results for a model *trained* on Astrid and *tested* on Astrid (top left), SIMBA (top middle), IllustrisTNG (top right), SB28 (second row left), Magneticum (second row right), and SWIFT-EAGLE (third row). The bottom panel shows the results of testing on CV sets of Astrid, SIMBA, IllustrisTNG, and Magneticum. *Source:* Reference (71).

the stellar mass, the galaxy radius, and the star metallicity. While the model of those authors was more precise than ours (likely due to the use of additional galaxy properties), it was not robust. However, our models are slightly more robust; we believe this could be due to the fact

that we use catalogs with different stellar mass thresholds to train the models, which overcomes the differences due to the fact that we are marginalizing over different stellar mass thresholds. This conclusion agrees with what authors from Reference (52) have found using the same idea of marginalization over an augmentation technique.

We reach similar conclusions when testing our models on galaxy catalogs from simulations of the CV sets (see the last panel of Figure 53), especially noticing that we have obtained a bias in the predictions for the different simulations. We emphasize the importance of testing the models on simulations as diverse as possible. Should we only have galaxy catalogs from Astrid and Magneticum simulations, we could reach the wrong conclusion that the model was both more precise and accurate than the one constructed using only positions and velocities.

6.1.6.3 Where does the information come from?

We now investigate where the information from our robust model (discussed in Section 6.1.6.1) comes from. Since in that model we only made use of galaxy positions and velocities, there are only three possibilities:

1. The information is coming from the positions of galaxies (clustering).
2. The information is coming from the distribution of galaxy velocities.
3. The information is coming from both positions and velocities.

Note that we are not considering attributing the importance to the level of information coming from the number of galaxies in the catalogs because: a) as mentioned in the Footnote 3, this global property only improved slightly the results, and b) we do not have a considerable number of catalogs with the same number, or even with the same range of the number, of galaxies (see Section 6.1.1.3). This last reason should result in worse predictions due to the lack of data to train the machinery and would not allow to test it in all the different subgrid physics models (which is the case of Magneticum, which only contains boxes with thousands of galaxies–see again Section 6.1.1.3).

In order to address the first possibility we have made use of graphs where the nodes do not contain any property. We train the model on Astrid, using the first slightly different GNN architecture described in Section 6.1.2.1: galaxy positions, i.e., using the prescription presented in Equations 3.20-3.21 for the first message passing layer. We then test the model on different graphs from different simulation suites. The results are presented in Figure 54, following the same scheme as Figure 47. In all the tests the results are visibly worse (with large error bars) and significantly biased (when extrapolating to the other simulations). More specifically, we found: $RMSE \in [0.084, 2.230]$, $R^2 \in -[0.680, 0.063]$, $PCC \in [-0.349, 0.854]$, $b \in [-0.5305, 0.0467]$, $\epsilon \in [24.3, 483.2]\%$, and $\chi^2 \in [9.957, 70.730]$. While testing the model in the CV sets we found a low performance for all the metrics analyzed, with larger error bars. Our

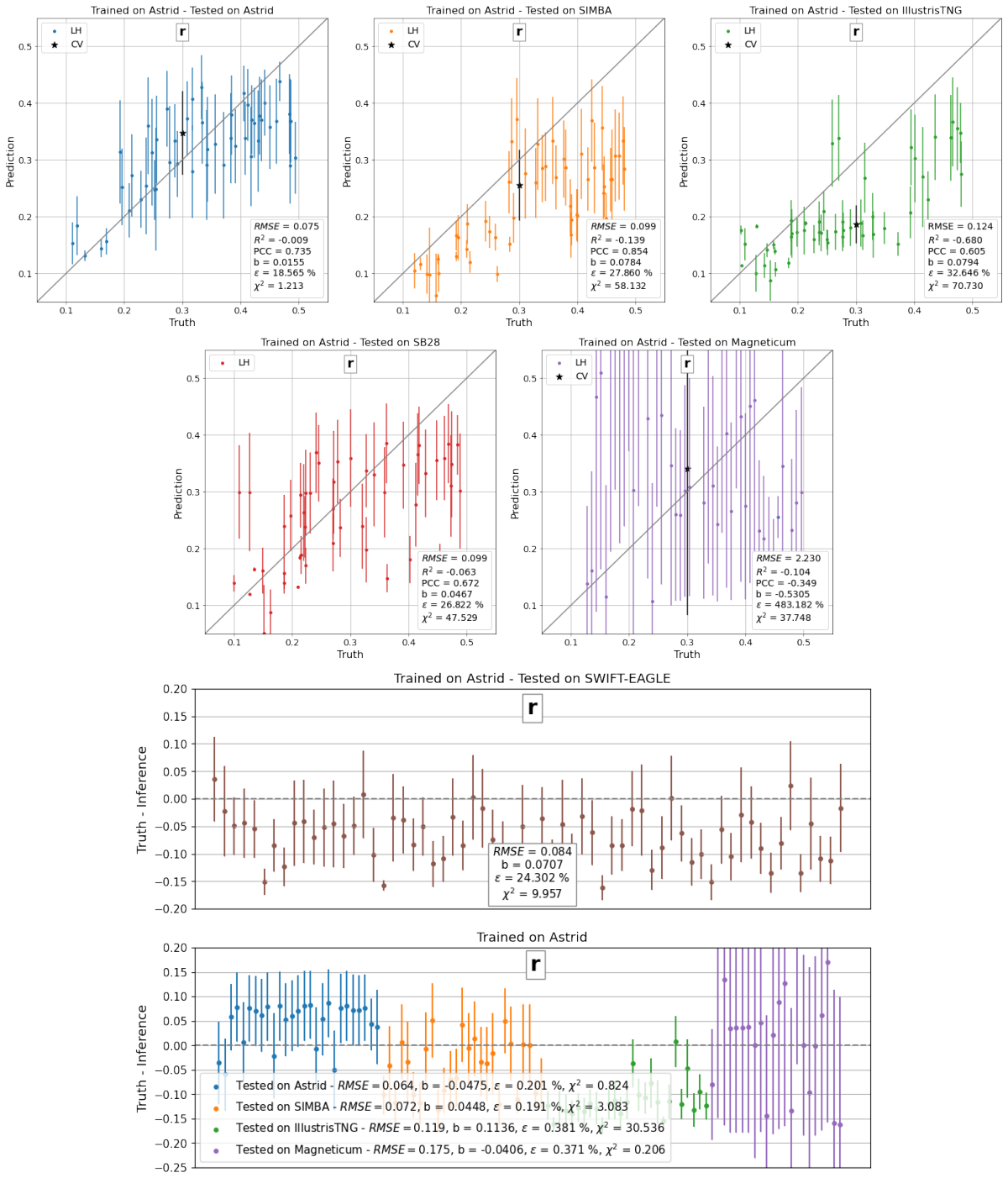

Figure 54 – **Ω_m predictions for a model trained on Astrid, using only galaxy positions.** We present the results for models *trained* on Astrid and tested on Astrid (top left), SIMBA (top middle), IllustrisTNG (top right), SB28 (second row left), Magneticum (second row right), and SWIFT-EAGLE (third row). The bottom panel shows the results of testing on CV sets of Astrid, SIMBA, IllustrisTNG, and Magneticum. *Source:* Reference (71).

results are qualitatively in agreement with those of Reference (50), who performed a similar analysis but with galaxy catalogs with a fixed stellar mass threshold and did not use Astrid as the training set. From this test, we conclude that the network cannot be extracting the information just from galaxy clustering.

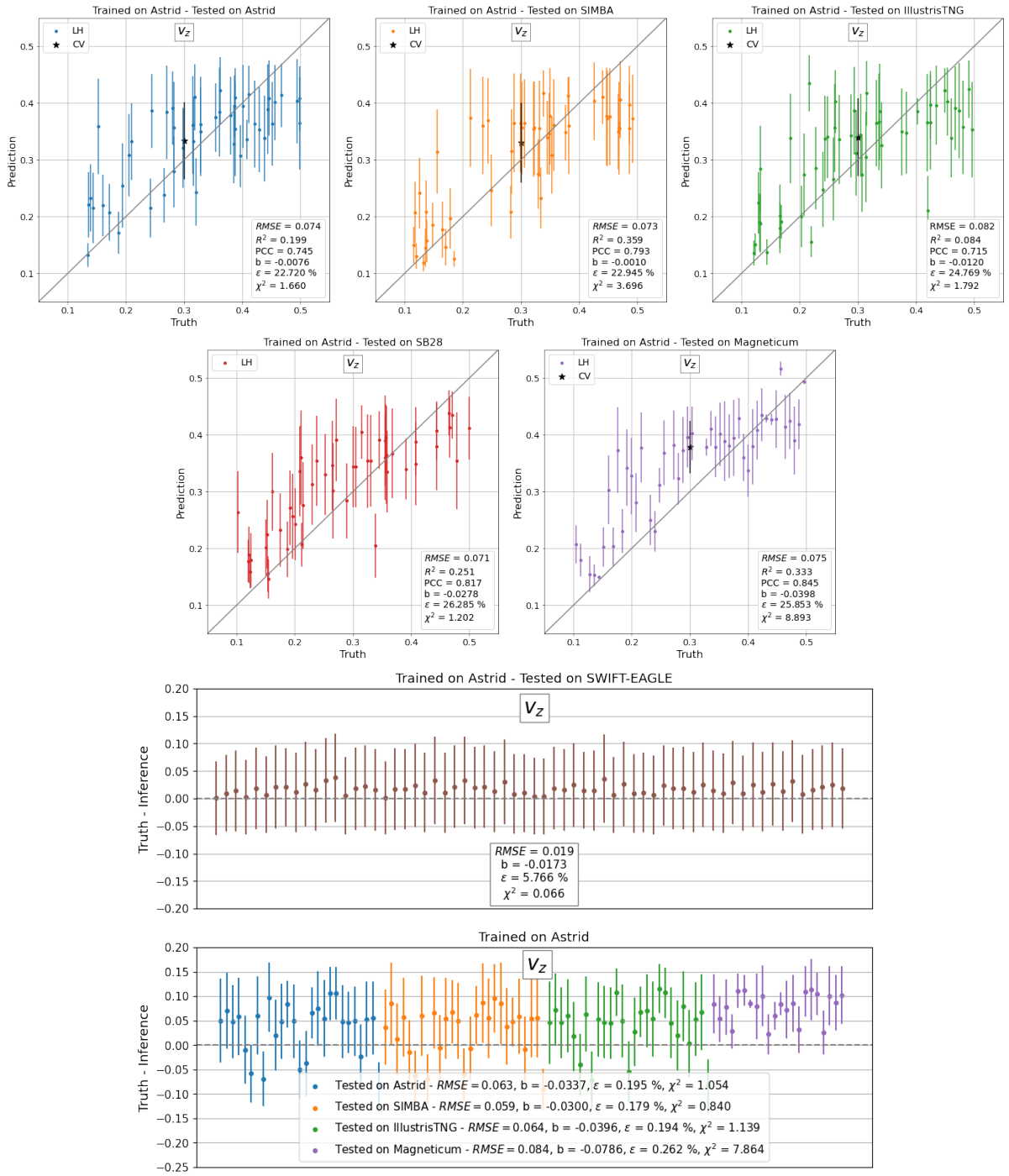

Figure 55 – **Ω_m predictions for a model trained on Astrid, using only galaxy velocities.** We present the results for a model *trained* on Astrid and *tested* on Astrid (top left), SIMBA (top middle), IllustrisTNG (top right), SB28 (second row left), Magneticum (second row right), and SWIFT-EAGLE (third row). The bottom panel shows the results of testing on CV sets of Astrid, SIMBA, IllustrisTNG, and Magneticum. *Source:* Reference (71).

Next, we train a deep set model (see the second model presented in Section 6.1.2.1: galaxy velocities) on galaxy catalogs that only contain the z component of the galaxy velocities (i.e. there are no galaxy positions) and, then, we employed Equation 3.19. We used Astrid simulation to train the model. Figure 55 displays the results. Also in this case we find that the

model performs poorly: $RMSE \in [0.019, 0.082]$, $R^2 \in [0.084, 0.359]$, $PCC \in [0.715, 0.845]$, $b \in -[0.0398, 0.0010]$, $\epsilon \in [5.8, 26.3]\%$, and $\chi^2 \in [0.066, 8.893]$. These results are distinct from what authors of Reference (50) found (while using a deep set as well), whose scores were comparable to the ones from the GNN. Note that those authors used more galaxy properties and we only use the 1D velocity component. The results for catalogs of the CV sets have large error bars and poor values for all the metrics. We then conclude that galaxy velocities can not be alone the origin of the information extracted by the network.

The above tests indicate that the network is making use of both positions and velocities to infer the value of Ω_m. Another important point to highlight is that the models trained on galaxy positions alone and galaxy velocities alone, although not very precise, seem to also not be robust. This may indicate that the model that uses galaxy positions and velocities may be extracting robust information due to constraints in phase space (e.g. the necessity to fulfill the continuity equation), directly encoding effective information on Ω_m.

6.1.6.3.1 Why the model trained on Astrid is so good?

We already have noticed that the model trained on Astrid (see Section 6.1.6.1.1), in comparison with the models trained on SIMBA and IllustrisTNG (see Section 6.1.6.1.2), using only galaxy phase-space information, presents better predictions. We commented about the number of galaxies per catalogs as one of the explanations for it in Sections 6.1.6.1.1 and 6.1.6.1.2, referring to Figure 46. In this section we will present some of other Astrid characteristics that can explain this success. For a complete discussion see Reference (311).

Compared to IllustrisTNG and SIMBA simulation suites in CAMELS, the fiducial model of Astrid features the mildest AGN feedback and predicts the least baryonic effect on the matter power spectrum. The training set of Astrid covers a broader variation in the galaxy populations (specially in the star formation rate density, due to A_{SN2}[10]) and the baryonic impact on the matter power spectrum compared to its TNG and SIMBA counterparts. This is clear by taking a look at Figure 56, where we present the comparison of the ratio of the matter power spectrum (left) and global star formation rate density (SFRD) of the 1P simulations (where only one parameter is varied and other parameters are fixed) for IllustrisTNG, SIMBA, and Astrid. These findings are a huge indicative which can make ML models trained on the Astrid suite exhibit better extrapolation performance when tested on other hydrodynamic simulation sets.

6.1.7 Why not inferring σ_8?

In this section, we present our efforts on trying to infer σ_8 using galaxy catalogs as graphs to feed GNN models. We made a sequence of tests of properties to include as node

[10] A_{SN2} parameter modulates the speed of hydrodynamically-decoupled galactic winds in all the simulation suites. Although the detailed model is different among the simulations, the star formation in Astrid turns out to be more sensitive to the SN wind speed while compared to TNG or SIMBA.

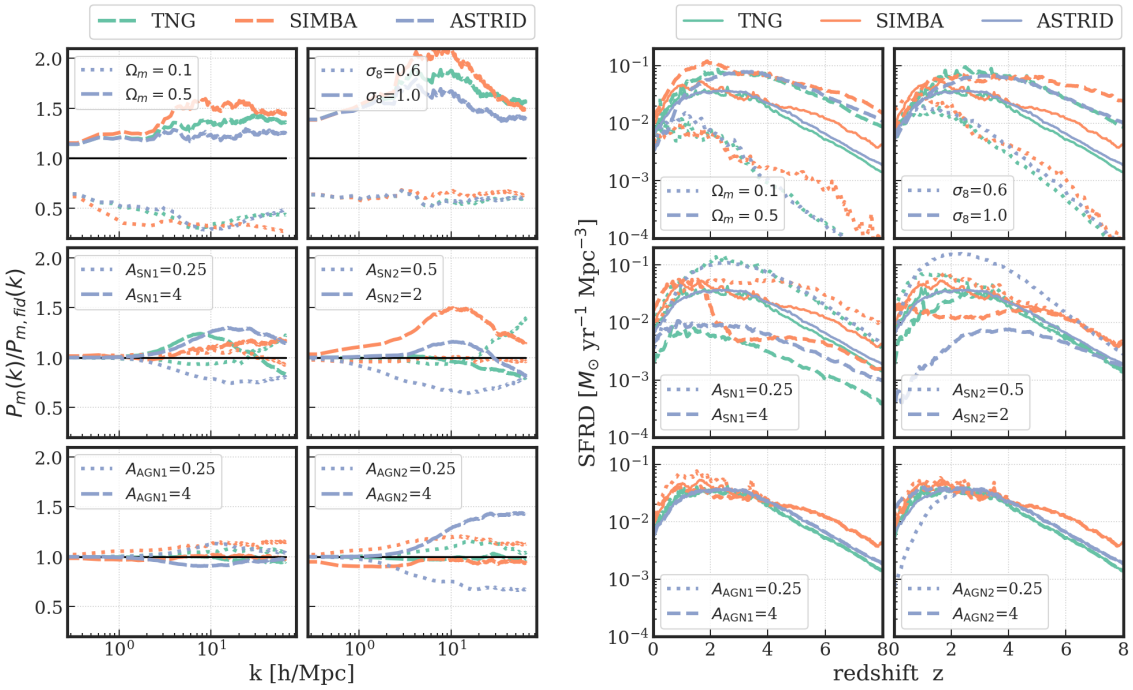

Figure 56 – **Ratio of the matter power spectrum (left) and global star formation rate density (SFRD) of the 1P simulations (where only one parameter is varied and other parameters are fixed).** Green, orange and blue colors represent the results from TNG, SIMBA, and Astrid, respectively. Solid, dotted, and dashed lines represent the simulations with the fiducial, lowest, and highest parameter values in the variation range correspondingly. *Source:* Reference (311).

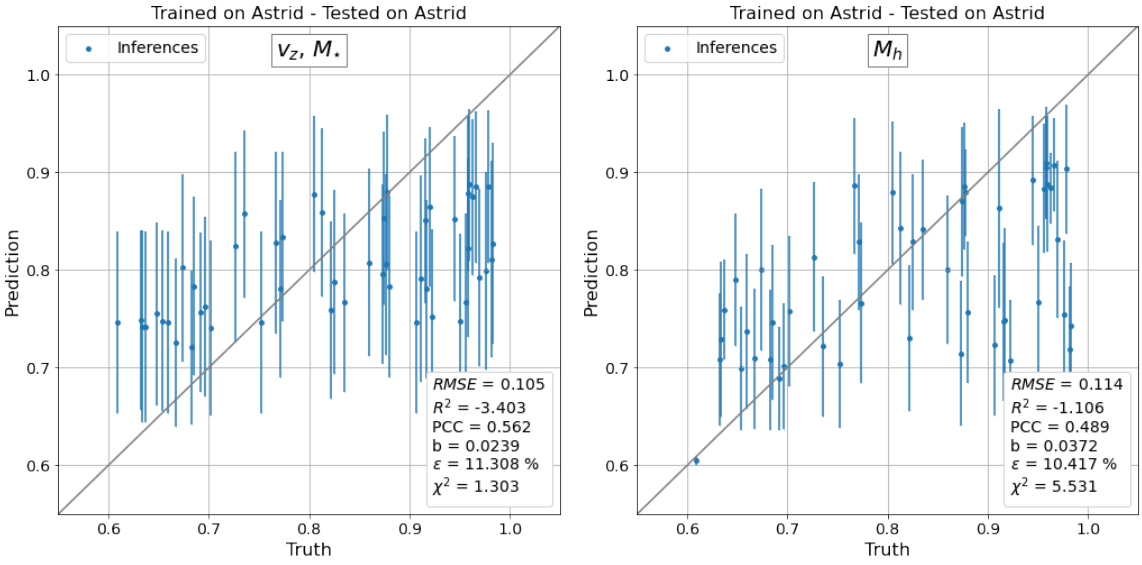

Figure 57 – **Predictions of σ_8 using galaxy and halo properties.** Likelihood-free inference of σ_8 using galaxy velocities on the z direction and stellar mass (on the left) and halo mass (on the right) as node attributes. We present the results for a model *trained* on Astrid and *tested* on Astrid. *Source:* Reference (71).

information in our graphs and none of them resulted in a robust model. Here we present two main results guided by:

1. Reference (50), while using galaxy velocities (in one direction) and including one more galaxy property, the stellar mass.

2. Reference (52), when using the host halo mass as node information for the graphs.

The results are shown in Figure 57. In both models, we found poor performance: higher values for RMSE (> 0.1), negative values for R^2 ($-[3.4, 1.1]$) and low values for PCC ($[0.49, 0.56]$). In the case of the model which uses the halo mass, the χ^2 value is higher too (> 5.5). Furthermore, the predictions are around the fiducial/mean value, without covering the whole range of values and having higher error bars.

As already shown by Reference (50), it is a challenge to infer this cosmological parameter using galaxy information, which may need more galaxy properties (stellar mass, galaxy radius, metallicity, and maximum circular velocity) to achieve better performance. Then, because of relying on galaxy properties that differ substantially among the different simulations, it is hard to get a robust model. That is why our inference while using only galaxy velocity and stellar mass, is worse than these authors' results. On the other hand, because we are using all galaxies (centrals and satellites), our results are not directly comparable to the ones presented in Reference (52), where only halos (without subhalos) are employed.

Therefore, we conclude, in agreement with References (50) and (70), that to constrain σ_8 precisely we need larger volumes, as no ML technique was able to infer their value using only galaxy information. Besides, getting the correct value of this parameter can be challenging also for the standard approaches due to the small size of the boxes in the CAMELS suite. One possible solution can be found in Reference (42), where the authors obtained good constraints to predict σ_8 using ML methods to deal with the usual summary statistics, for larger boxes ($100\ h^{-1}$Mpc). Another possible way to solve the puzzle related to σ_8 predictions should train a GNN on galaxy catalogs at higher redshifts and look for their impact on galaxy populations. This can be mostly related to the response of σ_8 in the abundance of more massive structures due to hierarchical structure formation, which does not happen at $z = 0$, where small galaxy populations dominate (311).

6.1.8 Discussion and conclusions

The quest to extract the maximum information from galaxy redshift surveys has motivated the development of many different approaches (32–40, 94, 344), and the upcoming data from the current and next generation of surveys (22–24, 26–31) is pressing this field of research. While we do not have a final answer to this question, ML techniques are promising tools that can be harnessed in order to tackle this problem (42, 44–49). In particular, GNNs stand out as good machinery to extract cosmological information from galaxy and halo catalogs from simulations (50–52, 228).

GNNs are ideal methods to analyze galaxy redshift surveys because: 1) they are designed to work with sparse and irregular data (219–221); 2) it is easy to construct models that fulfill physical symmetries (50); 3) they do not impose any cutoff scale. Perhaps the most challenging task associated with ML methods is their robustness (345), a hard question already explored using 2D maps with CNNs (48), tabular data (70), and galaxy catalogs (50). The reason behind the lack of robustness of the models is unclear and can be due to multiple factors: 1) data sets do not overlap; 2) models may be learning no physical effects (e.g. numerical artifacts); 3) data representation is different. We emphasize that precision is completely irrelevant without accuracy. The only way to deploy ML models to perform analysis with real data is to employ accurate models. Thus, robustness lies at the heart of this problem.

In this work, we have trained GNN models on thousands of galaxy catalogs from state-of-the-art hydrodynamic simulations of the CAMELS project to infer the value of Ω_m at the field-level using a likelihood-free approach. More importantly, we have investigated the robustness of the models by testing them on galaxy catalogs from simulations run with completely different codes to the ones used for training. We now outline the main takeaways from this work:

- The model trained on Astrid catalogs that only contain galaxy positions and velocities (the z component) is able to infer the value of Ω_m with $\sim 12\%$ precision and accuracy when tested on Astrid catalogs with different cosmologies and astrophysical parameters.

- The performance is similar when tested on galaxy catalogs from other galaxy formation simulations (each with different cosmology and astrophysics) run with four different hydrodynamic codes: IllustrisTNG, SIMBA, Magneticum, and SWIFT-EAGLE. This fact illustrates the robustness of the model under variations of the underlying subgrid physics.

- It also works well when tested on the SB28 set of the IllustrisTNG suite: a collection of $1,024$ simulations that varies 28 parameters (5 cosmological and 23 astrophysical) and therefore goes well beyond the diversity used to train the model (where only 6 parameters are varied).

- Our model is also robust to changes in the halo/subhalo finder: the galaxy catalogs of the SWIFT-EAGLE simulations were constructed employing VELOCIraptor, a different method than the one used for training (SubFind). When we tested our model on SWIFT-EAGLE catalogs we still obtained good predictions.

- The above constraints were obtained using a very small volume $(25\ h^{-1}\text{Mpc})^3$ that only contains $\sim 1,000$ galaxies with stellar masses above $\sim 2\times10^8\ M_\odot/h$ at $z=0$. We note that some galaxy catalogs contain a much larger ($\sim 5,000$, which is the case of Magneticum simulations) or smaller (~ 30, in some Astrid boxes) number of galaxies,

and the model trained on Astrid still performs well on those. This is related to the broader number of galaxies found on Astrid (the training set), together with other wide variety on this galaxy properties (311).

- When training our models on galaxy catalogs that contain positions, velocities, and stellar masses we are able to build models that are more precise but less accurate. In fact, those models are no longer robust across different simulation codes, and therefore could not be used with real data at the present time.
- We find that our models are extracting information from both galaxy positions and velocities. Furthermore, models trained using catalogs that only contain galaxy positions or galaxy velocities are not only less precise but also less accurate. We speculate that having both positions and velocities may improve the accuracy of the models as the phase-space distribution is constrained by physical arguments, such as the continuity equation, that need to be fulfilled independently of cosmology, astrophysics, and subgrid model employed.

Given the precision and accuracy of our model, it will be interesting applying it to peculiar velocity surveys such as the SLOAN catalog (346) or even the Cosmicflows-4 catalog (347). We note that several steps need to be carried out before performing such a task:

- The method needs to be shown robust with regards to super-sample covariance (fluctuations on scales larger than the simulation box) This is because in this analysis we did not account for such effect at the training stage. If the method is not robust to this effect, we should retrain our models on galaxy catalogs from larger volumes or catalogs, which would then suppress super-sample covariance. We note that preliminary work indicates that the models can deal with this effect, at least partially – see Section 6.1.6.1.3) for further details.
- Throughout this work we are dealing with peculiar velocities from simulations. Obviously, the peculiar velocities of galaxies cannot be measured with infinite precision. In particular, we do not take into account inaccuracies due to observational errors in this quantity. We therefore need to quantify how the errors on the peculiar velocities propagate into the constraint in Ω_m.
- An investigation on whether selection effects may affect the results is also needed, as most surveys rely on tracers that are not equally available at all redshifts.

The last two aforementioned topics have been the object of another study, as discussed in Section 6.3.

To summarize this Section, the work presented here is a new method to study Cosmology using the clustering and velocities of galaxies at the field-level, without imposing any cut on

scale, that seems robust to changes in cosmology, astrophysics, subgrid physics, and galaxy identification algorithms.

6.2 A universal equation to predict Ω_m from halo and galaxy catalogs

In this section we introduce a universal equation designed to predict Ω_m from halo and galaxy catalogs, based on Reference (310). These equations are formulated by integrating a GNN architecture with SR (see Section 3.2.3). The halo catalogs utilized for training this machinery are presented in Section 6.2.1, and data pre-processing methods (for halo and galaxy catalogs) are detailed in Section 6.2.1.1. Initially, the GNN is trained on DM halos from Gadget N-body simulations to perform field-level likelihood-free inference. The architecture of the GNN, along with its training and optimization processes, are described in Section 6.2.2. Subsequently, SR is employed to extract the optimal set of equations that encapsulate the behavior of the GNN, a procedure delineated in Section 6.2.3. The resultant GNN model demonstrates the capability to infer Ω_m with $\sim 6\%$ accuracy from halo catalogs of of N-body simulations run with six different codes. Furthermore, by applying SR to dissect the constituent components of GNN, we derive equations capable of predicting Ω_m from halo catalogs of simulations run with all of the above codes, achieving accuracies on par with those of the GNN. Moreover, we illustrate that by fine-tuning a single free parameter, our equations can extend their predictive capabilities to infer the value of Ω_m from galaxy catalogs sourced from hydrodynamic simulations of the CAMELS project. Although we present selected results in Section 6.2.4, we refer the Reference (310) for a complete analysis.

6.2.1 Data

We train our models using halo catalogs from high-resolution cosmological simulations that contain two halo properties: the halo positions **r** and the halo velocity modulus v. In this work we focus on halo and galaxy catalogs at $z = 0$.

The different N-body codes follow the evolution of DM particles under the effect of self-gravity in a given expanding cosmological background using different numerical techniques and approximations (see Section 2.5.1). The 6 codes we use to run the N-body simulations are described briefly below:

- **Abacus.** This code computes the long-range gravitational potential by decomposing the near-field and far-field forces in which the near-field forces are reduced to a r^{-2} summation (or an appropriately softened form) and the far-field forces to a discrete convolution over multipoles (348). We run 51 simulations with Abacus: 1 simulation with a shared cosmology and initial random seed among codes and 50 simulations in a LH with varying values of Ω_m and σ_8.

- **CUBEP^{3}M.** This code employs a particle-particle particle-mesh (P^{3}M) scheme, described in Reference (349), where long-range gravitational forces are computed via a 2-level particle mesh calculation. We ran 51 CUBEP^{3}M simulations: 1 simulation with shared cosmology and initial random seed among codes and 50 simulations in a LH.

- **Enzo.** This is an Adaptative Mesh Refinement (AMR) code, as described in Reference (350), that solves the Poisson equation via a fast Fourier technique (351) on the root grid and a multigrid solver on the individual sub-mesh. We only have one Enzo simulation which shares the same cosmology and initial random seed with the other codes.

- **Gadget.** This code utilizes a TreePM algorithm to compute short-range forces and Fourier techniques to calculate long-distance forces, as described in Reference (124). We use the halo catalogs from these simulations to train the models. We run $1,001$ of the Gadget simulations: 1 simulation with shared cosmology and initial random seed among codes and $1,000$ simulations that have different values of Ω_m, σ_8, and initial random seed.

- **PKDGrav3.** This code computes forces using Fast Multipole Method (FMM) (352, 353). We run $1,001$ N-body simulations with this code: 1 simulation with shared cosmology and initial random seed among codes and $1,000$ simulations with different values of Ω_m, σ_8, and initial random seed that are organized in a LH.

- **Ramses.** This code uses the Adaptive Particle Mesh technique described in Reference (354). It solves Poisson's equation level by level using Dirichlet boundary conditions and a Multigrid relaxation solver. We have run $1,001$ Ramses simulations: 1 simulation with shared cosmology and initial random seed among codes, and $1,000$ simulations with different values of Ω_m, σ_8, and initial random seed that are organized in a LH.

The different hydrodynamical simulations used to test the models presented in this work are the same of the ones described in Section 6.1.1.1.

6.2.1.1 Halo and galaxy catalogs

Here, we describe the procedures for constructing the halo and galaxy catalogs that we use to train, validate, and test the GNN and symbolic expressions.

- **Halo Catalogs for Training and Validating.** For training and validation, we use halo catalogs from the Gadget simulations. For each simulation, we generate 10 halo catalogs by taking all halos with masses larger than M_X, where M_X is a randomly chosen number between $100\,m_p$ and $500\,m_p$. Here, m_p is the mass of a single DM particle. As explained in Reference (52), using different DM particle thresholds is key to achieving a model that is robust to different simulations. These halo catalogs are generated by running ROCKSTAR (a halo/subhalo finder, see Section 2.5.1.8) (136) on snapshots from the numerical simulations described above.

- **Halo Catalogs for Testing.** We use all N-body simulations described in the previous section and 4 hydrodynamic simulations: IllustrisTNG, SIMBA, Astrid, and Magneticum. For each simulation, we generate 5 halo catalogs for the 5 different DM particle thresholds: $\{100, 200, 300, 400, 500\}$. Note that for hydrodynamic simulations, instead of considering only the amount of DM mass to make our mass cuts, we define m_p as the effective particle mass: $m_p = \frac{1}{N_c}\Omega_m V \rho_c$, where V is the volume of the simulation, ρ_c is the Universe's critical density today, and $N_c = 256^3$ is the effective number of particles. These halo catalogs are generated by running Rockstar (136) on snapshots from the numerical simulations described above.

- **Galaxy Catalogs for Testing.** We use galaxy catalogs from all the hydrodynamic simulations described in Section 6.1.1.1. We define a galaxy as a subhalo (can either be a central or satellite) that contains a stellar mass of at least $N \times m_*$ where $N \in 3, 4, 5, 6$ and $m_* = 1.3 \times 10^7\ h^{-1} M_\odot$. For each simulation, we construct 4 catalogs, each using a different N. We limit the range of the stellar mass thresholds to be no larger than $6 \times m_*$ because we find that using larger cuts result in catalogs with galaxy number densities that are smaller than the number densities (from the halo catalogs) used to train the network and equations. We find that using catalogs with number densities that are outside the training range can lead to inaccurate predictions. These galaxy catalogs are generated by running Rockstar (136) on snapshots from the 6 hydrodynamic simulations, with the exception of the catalogs from the SWIFT-EAGLE simulations which were generated using the halo finder VELOCIraptor (also see Section 2.5.1.8) (137, 138).

6.2.2 GNN architecture

The methods described in this section closely follow those presented in References (52, 71) to infer Ω_m. We emphasize the key changes that we implement in this work are: 1) using only the summation operator as the aggregation function and 2) reducing the depth and width of the GNN architecture with constrained hyperparameter optimization. These steps decrease the complexity of the model and allow for easier interpretation of the learned relations.

The halo and galaxy graphs were constructed in the same fashion way as presented in Sections 3.3.4.1 and 6.1.1.3, but using the 3D phase-space information and not accounting for the global attribute related to the number of objects. The architecture of the GNN follow the *message passing scheme* making use of a single MetaLayer (see Section 3.3.4.2.1) for a *compressed* GNN. For this reason, we denote the edge and node features that are input to the message-passing layer (the initial halo properties) with the superscript (0) and output (hidden) features by the message-passing layer with the superscript (1). Then:

- **Edge model:**

$$\mathbf{e}_{ij}^{(1)} = \mathcal{E}^{(1)}\left(\left[\mathbf{n}_i^{(0)}, \mathbf{n}_j^{(0)}, \mathbf{e}_{ij}^{(0)}\right]\right). \tag{6.17}$$

- **Node model:**

$$\mathbf{n}_i^{(1)} = \mathcal{N}^{(1)}\left(\left[\mathbf{n}_i^{(0)}, \sum_{j\in\mathfrak{N}_i} \mathbf{e}_{ij}^{(1)}\right]\right), \tag{6.18}$$

  where we reduce the aggregation function just to the summation to decrease the complexity of the learned relations.

The final layer (see Equation 6.10) in the architecture aggregates the hidden node features output by the message passing layer to make the prediction $\mathbf{y}$:

$$\mathbf{y} = \mathcal{F}\left(\left[\sum_{i\in\mathcal{G}} \mathbf{n}_i^{(1)}\right]\right). \tag{6.19}$$

Note that, different from the work presented in Section 6.1 and in Reference (71), we do not make use of the graph global attribute.

6.2.2.1 Training procedure and optimization

We train and test the models using graphs constructed from halo catalogs of the Gadget simulations. For each simulation, we construct 10 catalogs using the procedure described in Section 6.2.1.1 to marginalize over the halo number density. Once trained, the model is tested using catalogs from all simulations. For Gadget, we split the simulations into training (80%), validation (10%), and testing (10%) data sets before creating halo catalogs for each simulation. For the other codes, we use the entirety of the data set for testing.

We standardize the values of input node features as

$$\tilde{v} = \frac{v-\mu}{\delta}, \tag{6.20}$$

where μ and δ denote the mean and standard deviation of the feature $v = \sqrt{v_x^2+v_y^2+v_z^2}$. However, we explain in later sections that the value of δ must be tuned for when evaluating the symbolic equations.

Also, the GNNs are associated to the MNNs, making use of the loss function of Equation 3.25 (232), in the same fashion way as presented in Section 6.1.3. Our model is implemented in PyTorch (202) and PyTorch Geometric (222). We use the AdamW optimizer (355) with beta values equal to 0.9 and 0.999. We train the network using a batch size of 8 for 500 epochs.

The hyperparameters for our model are: 1) the learning rate, 2) the weight decay, and 3) the linking radius. We use the Optuna code (243) to perform Bayesian optimization and find the best value of these hyper-parameters for each model (see Section 3.5.4). Different from the previous work (see Section 6.1 and Reference (71)), the optimal linking radius was $\sim 1.35h^{-2}$ Mpc. As mentioned earlier, we aim to reduce the depth and width of our GNN architecture to obtain a *compressed* network, so we restrict to only one layer and 2 hidden neurons. For each model, we run 100 trials, where each trial consists of training the model using selected values

of the hyper-parameters. We perform the optimization of the hyper-parameters required to achieve the lowest validation loss possible and use early stopping to save only the model with a minimum validation error.

6.2.3 SR architecture

In this section, we describe the SR algorithm we use and the procedure for fitting functions to components of the learned GNN. We use the package PySR (199) (see Section 3.2.3). A key limitation of SR is that its tractability and accuracy are restricted to low-dimensional spaces of input data. To circumvent this, we limit the size of the latent space produced by the GNN, as described in Section 6.2.2. Using the learned parameters and relations from the low-dimensional GNN architecture, we search for equations that characterize the model by approximating the individual MLPs used in the node model, edge model, and final layer described in Equations 6.17, 6.18, and 6.19, respectively. We emphasize that since there is only one message-passing layer, we only need to approximate one node model MLP and one edge model MLP. Moreover, for each of the node and edge models, we search for two equations because there are two hidden features.

The data and procedure used to obtain these equations are described below:

- **Approximating Edge Model.** To approximate the edge model, we train a SR to map from the input variables $\mathbf{x}^e$, to the target variables $\mathbf{y}^e$, defined as:

$$\mathbf{x}^e = \left(n_i^{(0)}, n_j^{(0)}, \alpha_{ij}, \beta_{ij}, \gamma_{ij}\right) \tag{6.21}$$

$$\mathbf{y}^e = \left(e_1^{(1)}, e_2^{(1)}\right). \tag{6.22}$$

  The input variables are the initial features of the nodes and their neighbors, as well as the initial edge features as described in Section 6.2.2.1. The corresponding target variables are the edge features of the MLP in the edge model defined in Equation 6.17. Since the GNN employs only two hidden features for each message-passing layer, we denote the first component of the edge feature as $e_1^{(1)}$ and the second component as $e_2^{(1)}$. To obtain this data, we randomly select 10 $(\mathbf{x}^e, \mathbf{y}^e)$ pairs from each graph in the training set. This selection is done to ensure that we have a representative sample of the training set without using every node pair of all graphs which would result in too large of a data set.

- **Approximating Node Model.** Similarly, to approximate the node model, the input variables $\mathbf{x}^n$, and the target variables $\mathbf{y}^n$, of the symbolic regressor, are:

$$\mathbf{x}^n = \left(n_i^{(0)}, \sum_{j \in \mathcal{N}_i} e_1^{(1)}, \sum_{j \in \mathcal{N}_i} e_2^{(1)}\right) \tag{6.23}$$

$$\mathbf{y}^n = \left(n_1^{(1)}, n_1^{(1)} + n_2^{(1)}\right). \tag{6.24}$$

As seen above, the inputs are the initial node feature and the neighborhood-wise sums of the hidden edge features because the output of the edge model is aggregated using the summation operator before being passed onto the node model. The corresponding target variables are the hidden node features of the MLP in the node model defined in Equation 6.18. We denote the first and second hidden node features as $n_1^{(1)}$ and $n_2^{(1)}$, respectively. However, instead of directly finding an equation for the second node feature, $n_2^{(1)}$, we instead search for a formula for the sum $n_1^{(1)} + n_2^{(1)}$. This is because we find that the change of variables allows us to obtain more accurate approximations than with the original target variable. Ultimately, to obtain the expression of $n_2^{(1)}$, we subtract from it $n_1^{(1)}$. To obtain this data, we randomly sample 10 $(\mathbf{x}^n, \mathbf{y}^n)$ pairs from each graph in the training set as we did with the edge model data.

- **Approximating Final MLP.** Lastly, to approximate the MLP in the final aggregation layer, the input and target variables are:

$$\mathbf{x}^u = \left(\sum_{i \in \mathcal{G}} n_1^{(1)}, \sum_{i \in \mathcal{G}} n_2^{(1)} \right) \tag{6.25}$$

$$\mathbf{y}^u = \mu_i. \tag{6.26}$$

Here, the inputs are the graph-wise sums of the hidden node features because the output of the node model is aggregated using the summation operator before being passed onto the final MLP. The corresponding target is the mean posterior (here, because we predict Ω_m, it is denoted as μ_{Ω_m}). We do not attempt to find an expression for the posterior standard deviation as it is solely a component of the parameter inference methodology and does not contribute additional physical understanding. We obtain this data from each graph in the training set. Note that this time there is no need to select a sub-sample of nodes from each graph because x^u and y^u are global properties of the graph so we can use every graph in the training set.

In each of the above approximation steps, the SR algorithm searches for analytic expressions that can map from the given input variables to the desired target. For the training, the regressor is allowed to employ the following operators: "ADD", "SUB", "MULT", "DIV", "POW", "1/X", "ABS", "LOG", "LOG10", "SQRT". We employ a standard MSE loss function to optimize the model too. The model was trained for $100,000$ trials with a batch size of 64.

During training, the algorithm outputs a list of equations found by the regressor (see Section 3.2.3) for each of the GNN components. We evaluate several candidate equations on a test set for each hidden feature before selecting one that optimizes the trade-off between complexity and accuracy with these metrics in mind. Finally, the performance metrics were the same as presented in Section 6.1.5.

We emphasize that the predictions of Ω_m derived from the SR equations incorporate all the equations responsible for updating both the edge and node models, in addition to

approximating the final MLP (which gives the final prediction for Ω_m, here denoted as μ_{Ω_m}). These equations are employed to substitute their respective MLP counterparts within the GNN model (node and edge models, as well as the final aggregation layer), while retaining all other components of the GNN unaltered.

Table 8 – **Analytic formulae obtained using SR for each component of the learned GNN model.** To the edge model, node model, and the MLP in the final aggregation layer. The last column lists the RMSE values of the analytic expressions when they are individually substituted into the GNN architecture. This evaluation is done by replacing the corresponding MLP in the edge model, node model, or final aggregation layer with the symbolic approximation while keeping all other components of the GNN unchanged. Notice that, the feature of node i is defined as $n_i = (|\vec{v}_i| - \mu)/\delta$, where $|\vec{v}_i|$ is the velocity modulus of halo/galaxy i, $\mu = 189$ km s^{-1}, and δ is a free parameter with units of km s^{-1} that needs to be adjusted for galaxy catalogs (see Section 6.2.2.1 and Table 9 for more details).

GNN Component	Formula	RMSE
Edge Model: $e_1^{(1)}$	$1.32\|n_i - n_j + 0.21\| + 0.12(n_i - n_j) - 0.12(\gamma_{ij} + \beta_{ij} - 1.73)$	0.03
Edge Model: $e_2^{(1)}$	$\|1.62(n_i - n_j) + 0.45\| + 1.98(n_i - n_j) + 0.55$	0.04
Node Model: $n_1^{(1)}$	$1.21^{n_i}(0.77^{3.29\sum_{j\in\mathcal{N}_j} e_1^{(1)} + \sum_{j\in\mathcal{N}_j} e_2^{(1)}}) + 0.12$	0.02
Node Model: $n_1^{(1)} + n_2^{(1)}$	$0.78 - \sqrt{\log(0.16^{\sum_{j\in\mathcal{N}_j} e_2 + \sum_{j\in\mathcal{N}_j} e_1 - 0.41 n_i - 1.05})} + 1.45$	0.03
Final MLP: μ_{Ω_m}	$4\times10^{-4}\cdot(-5.5\sum_{i\in\mathcal{G}} n_2^{(1)} + 2.21\sum_{i\in\mathcal{G}} n_1^{(1)} + \|0.96\sum_{i\in\mathcal{G}} n_2^{(1)} + 0.82\sum_{i\in\mathcal{G}} n_1^{(1)}\|) - 0.103$	0.03

6.2.4 Results

In this section we present some selected results from Reference (310). Firstly, we show the analytic approximations that were found using SR in Section 6.2.4.1. Secondly, in Section 6.2.4.2, we compare the predictions of GNNs and the resulting equations on halo catalogs. Thirdly, we present the predictions for Ω_m on galaxy catalogs for both methods, showing that, while GNNs fail on their predictions, the symbolic expressions are able to extrapolate their results (Section 6.2.4.3). A complete analysis can be found in Reference (310).

6.2.4.1 Analytic Approximations

Here we present the equations extracted from the trained GNN model using the SR method. The formulae for each of the hidden edge and node features, as well as for the predicted posterior mean from the final MLP, are listed in Table 8. The listed RMSE values are computed by individually replacing the corresponding component in the GNN architecture with each expression while keeping all other components of the GNN unchanged and evaluating them on halo catalogs of the Gadget test set. The computed RMSE values are used to gauge the error that each approximate equation introduces.

It is important to note that the variables n_i and n_j in the equations represent the initial node features or velocity moduli. As explained in Section 6.2.2.1, these variables were

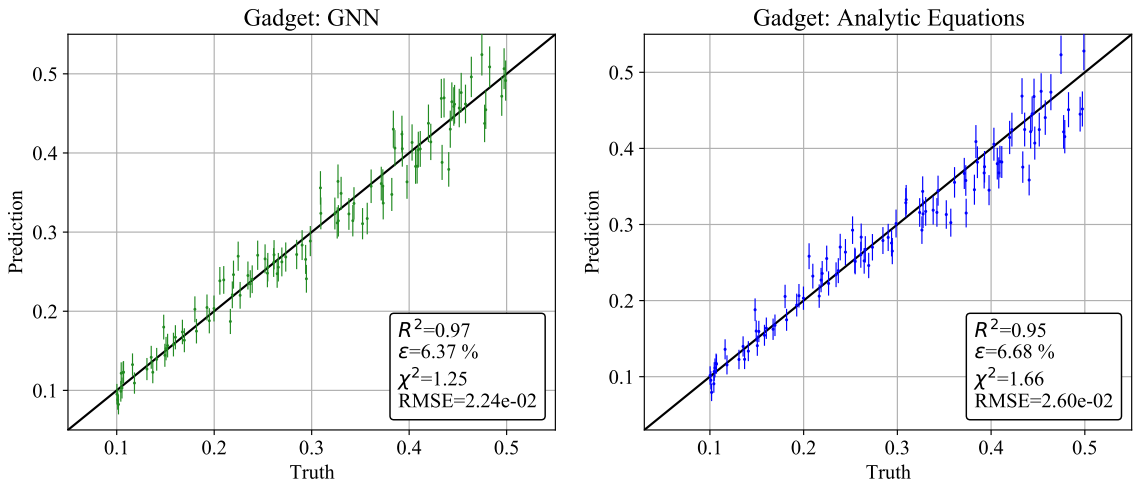

Figure 58 – **Comparison of Gadget $\Omega_{\rm m}$ predictions for the GNN and SR on halos.** We present GNN (on the left panel) and SR (on the right panel) predictions. *Source:* Reference (310).

normalized by the mean and standard deviation of the velocity modulus for the halos from the training set to ensure that all terms in the equations are dimensionless. Hence, the velocity modulus terms in the equations are $n_i = \frac{v_i - \mu}{\delta}$ and $n_j = \frac{v_j - \mu}{\delta}$, where $\mu = 189$ kms^{-1} is a fixed value that was the computed mean velocity modulus for all halos in the training set and δ is treated as a free parameter. For testing on halo catalogs, we set $\delta = 129$ kms^{-1} which is equal to the value used during training and was the standard deviation computed for all halos in the training set. On the other hand, for testing on galaxy catalogs, we tune δ to fit to each hydrodynamic simulation set as listed in Section 6.2.2.1 because we find that using the value $\delta = 129$ kms^{-1} leads to inaccurate predictions. This is not surprising given that this value was computed for N-body halos which would not be expected to extrapolate to galaxies. Hence, it is possible that tuning it for different simulations can account for the halo-galaxy bias. We discuss this in more detail in Section 6.2.5.

The way to use these equations is as follows. First, given a halo/galaxy catalog, a mathematical graph is constructed by considering the halos/galaxies as nodes and linking nodes by edges if their distance is smaller than $r_{link} = 1.35\ h^{-1}$Mpc (see Section 6.2.2.1 for details). For the graph, the feature of node i is defined as $n_i = (|\vec{v}_i| - \mu)/\delta$, where $|\vec{v}_i|$ is the velocity modulus of halo/galaxy i, $\mu = 189$ km s^{-1}, and δ is a free parameter with units of km s^{-1} that needs to be adjusted for galaxy catalogs (see Section 6.2.2.1 and Table 9 for more details). Also, the edge features β_{ij} and γ_{ij} between nodes i and j are computed using Equations 6.7 and 6.8, respectively. At this stage the graph is ready to feed the GNN, with their components (node and edge models and final aggregation layer) replaced by the equations presented in Table 8. Second, the graph will have their edge features updated using the first two equations of the edge model. Third, the updated node features are computed using the two equations of the node model. Finally, from the updated graph we can estimate Ω_m by using the final equation (referred as "final MLP" in Table 8).

The accuracy of the equations when evaluated on the halo catalogs of the Gadget simulations is shown in the right panel of Figure 58. As can be seen, these analytic approximations achieve similar mean relative error (6.7%) and RMSE (2.6×10^{-2}) as the GNN, suggesting that they are accurate representations of the trained network. We emphasize that our analytic formula predicts the posterior mean while the error bars (posterior standard deviation) are obtained from the GNN discussed in Section 6.2.3.

6.2.4.2 Predictions on halo catalogs

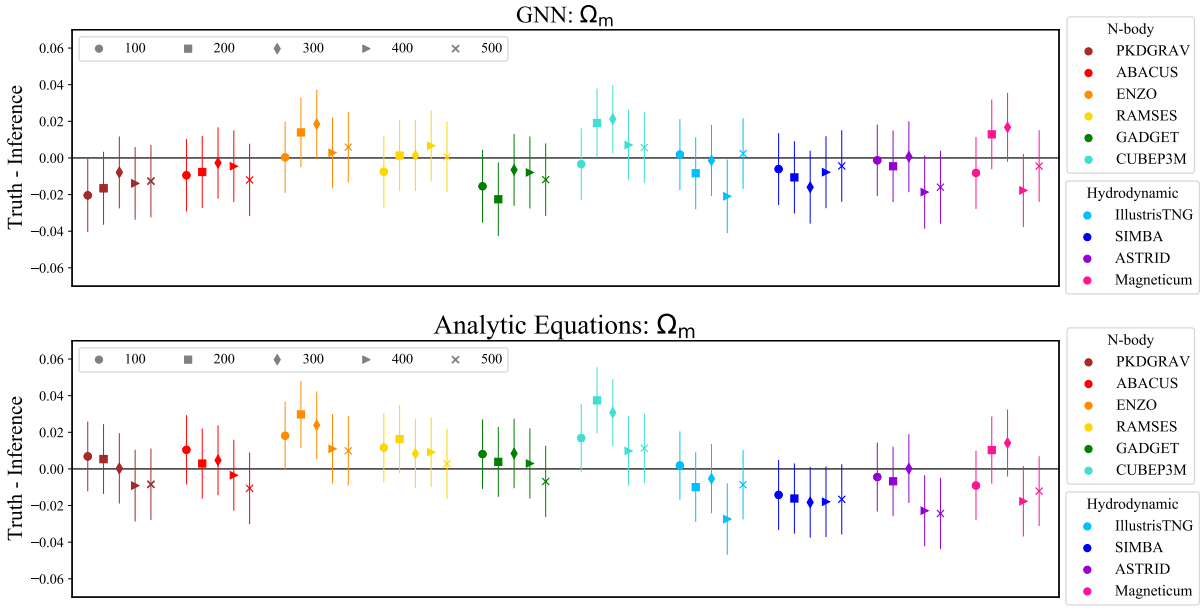

Figure 59 – **Ω_m predictions using GNNs and SR equations.** We present the GNN (on top panel) and SR equations (on bottom panel) predictions for halo catalogs of N-body (PKDGRAV, ABACUS, ENZO, RAMSES, GADGET, and CUBEP3M) and hydrodynamic (IllustrisTNG, SIMBA, Astrid, and Magneticum) simulations. Note that, for each simulation, we generate 5 catalogs. Each halo catalog contains all halos with masses above Nm_p, where m_p is the particle mass and N can be $100, 200, 300, 400$, or 500 (see legend). The y-axis represents the difference between the truth and the inference. *Source:* Reference (310).

In Figure 59 we present the comparison of the GNN and SR equation predictions of Ω_m on halo catalogs of N-body (PKDGRAV, ABACUS, ENZO, RAMSES, GADGET, and CUBEP3M) and hydrodynamic (IllustrisTNG, SIMBA, Astrid, and Magneticum) simulations. Note that, for each simulation, we generate 5 catalogs. Each halo catalog contains all halos with masses above Nm_p, where m_p is the particle mass and N can be $100, 200, 300, 400$, or 500 (see legend). The y-axis represents the difference between the truth and the inference.

As can be seen, both models exhibits surprising extrapolation properties and are robust to all simulation codes. This is remarkable to be on side of Occam's razor, because the GNN is on its compressed version, containing only one message-passing layer and agrees with the results from Reference (52) (which contains a more complex GNN). Also, the formulae maintain

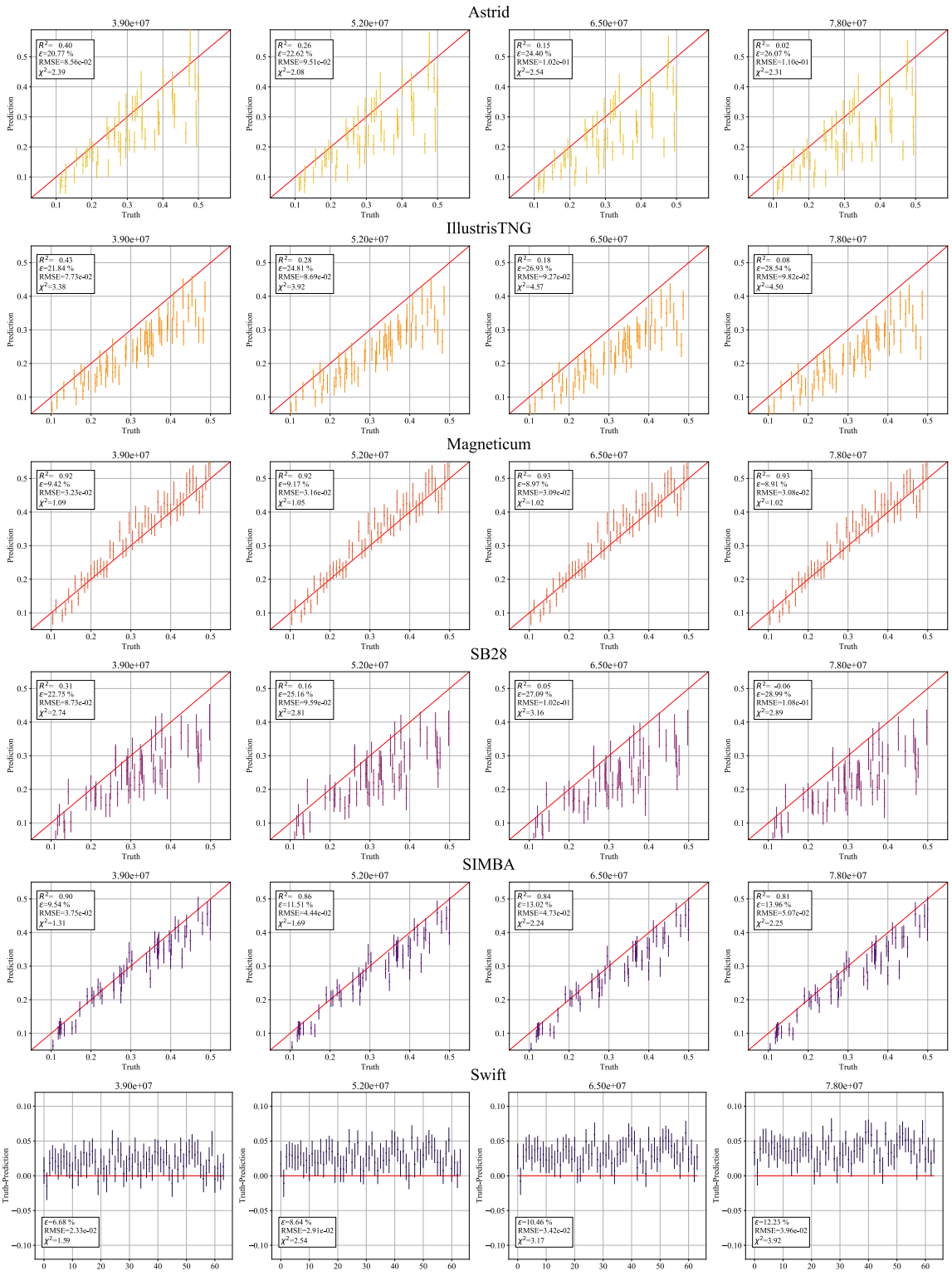

Figure 60 – **GNN (trained on halos) predictions on galaxy catalogs.** This plot shows the predictions of the GNN trained on halo catalogs from Gadget being tested on galaxies from 6 different hydrodynamic simulations. To construct the galaxy catalogs we use 4 different stellar mass thresholds which are labeled for each column. For clarity, we plot the predictions for 50 randomly selected catalogs in each panel, but the metrics are for the whole sets. *Source:* Reference (310).

the robustness of the GNN model and achieve a very similar accuracy compared to the GNN, deviating a bit more for halos from the hydrodynamical simulations.

6.2.4.3 Predictions on galaxy catalogs

We also test the GNNs and the SR equations on galaxy catalogs from the 6 hydrodynamic simulation suites: Astrid, IlustrisTNG, Magneticum, SB28, SIMBA, and SWIFT-EAGLE. We emphasize that this is not a trivial task as the GNN and the corresponding equations were trained using DM halos from N-body simulations that do not contain any information about the intergalactic dynamics or baryonic processes present in hydrodynamic simulations. There is also a complex halo-galaxy connection which can, for instance, be reflected in the relative abundances of halos and galaxies where larger halos can contain multiple galaxies while smaller halos may not contain any. These biases can possibly leave a significant imprint in the relations between the relative position and velocity terms of the equations found for halos. For these tests, we follow the definitions of galaxies and stellar mass thresholds discussed in Section 6.2.1.1 in constructing the galaxy catalogs where we include both central and satellite galaxies.

In Figure 60 we present the predictions of the GNN trained on halo catalogs and tested on galaxies from 6 different hydrodynamic simulation suites: Astrid, IllustrisTNG, Magneticum, SB28, SIMBA, and SWIFT-EAGLE. As it can be seen, the GNN is unable to accurately predict the values of Ω_m as all the predictions exhibit a bias deviating from the true values (no matter the cut considered on stellar mass). This is common across all simulations, which is expected given that there is a nontrivial connection between halo and galaxy distributions.

Table 9 – **List of the optimized values of δ.** We list the values for Astrid, IllustrisTNG, Magneticum, SB28, SIMBA, and SWIFT-EAGLE. We also include the δ used for testing on N-body halos for comparison.

Simulation	δ [km/s]
N-body codes	129.2
SB28	100.0
Astrid	126.5
SIMBA	122.5
IllustrisTNG	99.6
SWIFT-EAGLE	114.5
Magneticum	147.2

We believe that these biases are due to the effects of the halo-galaxy connection in addition to the differences in the abundance of galaxies found in the catalogs used for testing and that of halos found in the training data set. As discussed in Section 6.2.1.1, the network is unable to extrapolate to number densities outside of the training range and there are many catalogs with galaxy number densities that fall below the range of the halo number densities seen during training: $(1,000; 6,000)$. However, the under-predicted values of Ω_m cannot be

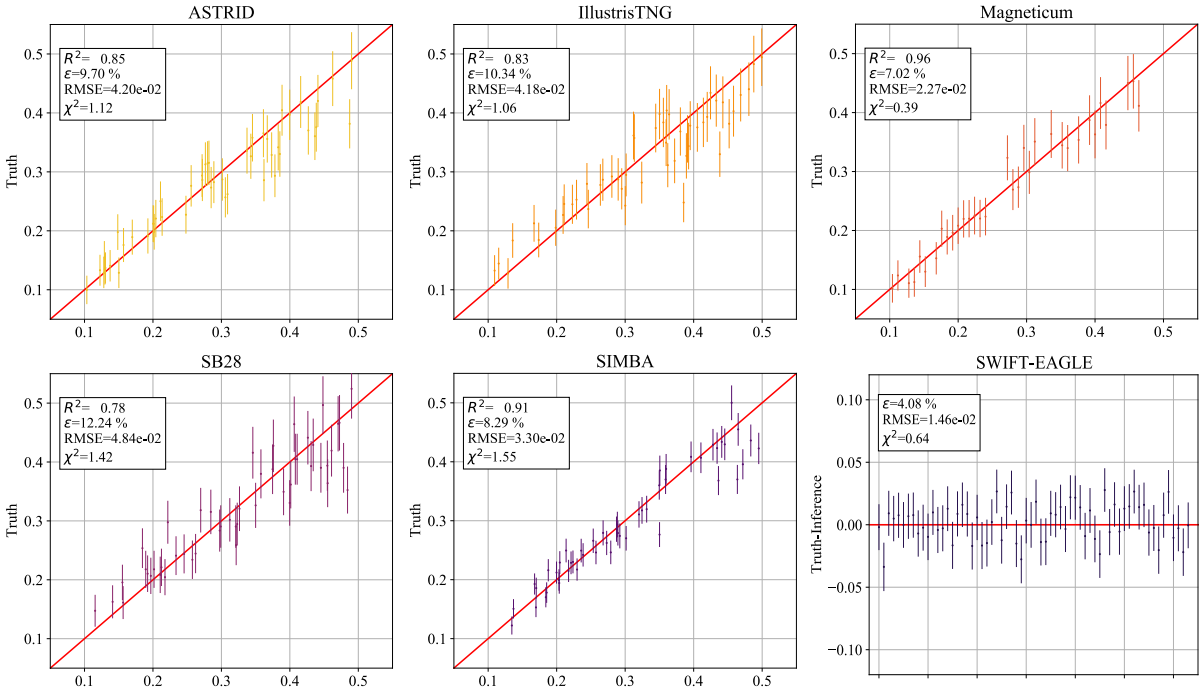

Figure 61 – **Tuned SR equations (trained on GNN blocks from halo catalogs) predictions on galaxy catalogs.** The predictions are done on 6 different hydrodynamic simulation sets: Astrid, IllustrisTNG, Magneticum, SB28, SIMBA, and SWIFT-EAGLE, to infer the value of Ω_m. We plot the predicted against truth for each simulation. We present results for a stellar mass threshold of $4 \times m_*$, with $m_* = 1.3 \times 10^7\ h^{-1} M_\odot$ (see Section 6.2.1.1), but we reach similar accuracies for other mass cuts. We also include only 50 randomly selected catalogs for each simulation set for the clarity of the figures, but the reported metrics were computed for all simulations in the suites. Note that for the bottom right panel, which depicts the predictions for the SWIFT-EAGLE simulation set, we use simulations that are generated with the same value of $\Omega_m = 0.3$. Thus, we plot the difference between the truth and the prediction on the y-axis for these catalogs. *Source:* Reference (310).

solely attributed to the abundance of galaxies. As discussed in the Reference (71) (see Figure 46, in Section 6.1.1.3), the full range of galaxy number densities is exhibited for all values of Ω_m, specially for Astrid and there are other galaxy properties in these simulations which cause differences to the galaxy properties. This agrees further demonstrates that the biases present in the network predictions are attributed to the intrinsic characteristics of the galaxy population.

We present the results for evaluating the equations on galaxy catalogs from the different hydrodynamic simulations in Figure 61. Each panel is labeled with the corresponding simulation suite. For simplicity, we present the predictions for only the galaxy catalogs generated with the stellar mass threshold of $4 \times m_*$ for a fixed m_* denoting the mass of an individual stellar particle as described in Section 6.2.1.1. However, we find that the equations are able to perform with similar accuracies for catalogs constructed with different mass cuts. Moreover, since the simulations from the SWIFT-EAGLE suite are run with the same value of Ω_m, we plot the difference between the true ($\Omega_m = 0.3$) and the predicted values on the y-axis for these catalogs.

We note that the presented error bars for all simulations are the inferred posterior standard deviation values obtained by the model trained and tested on galaxy catalogs discussed in Section 6.1 and in Reference (71), since the equations predict only the first moment of the posterior for Ω_m.

For each simulation we tune the parameter δ to improve the accuracy of the predictions. As discussed in Section 6.2.4.1, this parameter appears in the equation as a normalization of the velocity modulus, and its value varies for different hydrodynamic simulations when testing on galaxies. We tune this normalization because we noticed that using the original value $\delta = 129$ kms^{-1}, the standard deviation of the velocity moduli for all halos in the training set, resulted in predictions that deviated from the truth in terms of a slope and bias, which varies for each simulation. Thus, in Table 9 we list the values of δ that we optimize for each simulation using *nonlinear least squares* with SCIPY-OPTIMIZE[11] for the catalogs constructed using the same stellar mass threshold as presented here.

After tuning this parameter, we find that the equations are able to predict Ω_m with mean relative errors of 15.35% for ASTRID, 12.85% for ILLUSTRISTNG, 6.89% for MAGNETICUM, 16.17% for SB28, 8.50% for SIMBA, and 4.08% for SWIFT-EAGLE, across the four stellar mass thresholds. Evidently, the predictions for the galaxy catalogs exhibit significantly larger error than for the halo catalogs. This can be explained by two reasons. One, there are additional astrophysical processes and dynamics present in the thousands of hydrodynamic simulations that can interfere with the equations' extrapolation ability. Given that the equations can only encode information regarding the gravitational interactions between halos from N-body simulations, the effects of these various astrophysical parameters may impede on the accuracy of the predictions. Moreover, there are likely to be significantly more outliers for simulations such as SB28, where we vary 28 cosmological and astrophysical parameters at a time. This is also true for the ASTRID simulations which encompass a wider range of galaxy properties and are able to encapsulate the variations found in the other simulation suites.

Two, there is a large fraction of the galaxy catalogs that contain galaxy number densities outside the scope of the halo number densities seen by the GNN and equations during training. For instance, the number of halos in catalogs from the Gadget simulations used for training ranges from $\sim 1,000$ to $6,000$. However, there are galaxy catalogs that contain fewer than 500 galaxies at this stellar mass threshold. These outliers are particularly dominant in the ILLUSTRISTNG, ASTRID, and SB28 simulations, which leads to under-predicted values of Ω_m. On the other hand, if one removes these outliers, the mean relative errors significantly decrease. Hence, Figure 61 depicts the results for only the catalogs with galaxy number densities that fall within the range of $(1,000; 6,000)$. Restricting to these catalogs decreases the mean relative errors to: 9.76% for ASTRID, 10.34% for ILLUSTRISTNG, 7.02% for MAGNETICUM, 12.24% for SB28, 8.29% for SIMBA, and 4.08% for SWIFT-EAGLE.

[11] https://docs.scipy.org/doc/scipy/reference/optimize.html

Thus, we conclude that the equations are able to extrapolate to galaxies, with accuracies that are comparable to those attained for the halo catalogs from hydrodynamic simulations. These results are also comparable to those obtained by us in Reference (71) (see Section 6.1), where we trained a model directly on galaxy properties. We note that the effect of the number density being an uninformative prior during the learning process can be diminished by broadening the range of halo number densities used to train the network and equations.

6.2.5 Discussion and Conclusions

Here, we discuss some speculative interpretations of the equations that we found. In that respect, we only attempt to explain the formulae for the *edge models*, because their functional forms are simpler than those for the node models (as they introduce nonlinearities to the formulae). The edge model also solely employs physical information about the halo positions and velocity moduli, so they are responsible for directly leveraging the clustering and distribution of the halos. Also, we discuss the normalization choice for the velocities.

- **Relative Peculiar Velocity Modulus ($v_i - v_j$).** This indicates that the model is taking advantage of the relative velocity moduli of the halos and their neighbors. We believe that in this case, using the relative velocities allows the models to gauge the local gravitational forces where the relative velocity moduli between two halos can serve as a proxy for the depth of the potential wells in the bound system. This is reasonable since larger relative speeds of interacting bodies can result from the presence of stronger attractive forces between them. From this, the model may be learning a representation of the masses of the halos. An analogous discussion in Reference (308) reached similar conclusions pertaining to the pairwise peculiar velocities and speeds which were found to have strong dependence on Ω_m at the same small scale as that used by the models in this work ($\lesssim 5\ h^{-1}$Mpc). We also speculate that the presence of these terms reflect the strong dependence of Ω_m on the information available in the cosmic velocity fields (304, 356, 357). For instance, the authors of Reference (356) discuss a derived relation between the moments of the scalar field of the peculiar velocity divergence and Ω_m, which is independent of the biasing between the distribution of galaxies and the underlying DM density field. It is possible that the expressions found in this work reflect a similar relationship, because our models have been trained using the scalar halo velocity modulus, and the fact that they demonstrate an accuracy that is not significantly affected by the presence of astrophysical and baryonic effects. We speculate that the network and equations may be correcting for the nonlinearities of the galaxy velocity fields on smaller scales, by considering the galaxy distribution and number densities.

- **Velocity normalization.** Here we discuss the implications of tuning the normalization of the velocity modulus terms, δ, for galaxies from each simulation set. Previous findings in References (358,359) indicate that the halo-galaxy distribution bias can induce biases in

pairwise velocity statistics defined using the radial separation between galaxies. Thus, we speculate that the normalization of the velocity modulus terms v_i and v_j in our equations reflects a similar correction to account for the fact that the spatial clustering of galaxies may not trace directly the matter field.

- **Spatial distribution and clustering.** In the first edge equation, e_1, the presence of the terms β and γ reflects the spatial distribution of the halos in the catalogs. Specifically, the variable $\gamma \in (0, 1]$ describes the distance between two halos where its range is restricted due to its normalization by the linking radius, $r_{link} \sim 1.35\, h^{-1}$Mpc, as described in 6.2.2.1. Thus, a smaller γ would indicate a denser distribution of halos. Meanwhile, the variable $\beta \in [-1, 1]$ describes the angular orientation of a halo with respect to its neighbor, and can provide information about the shape of the distribution, e.g. the filamentary structure of the cosmic web. Both parameters are used by the model to learn about the presence of large scale structures such as superclusters and filaments.

Finally, we can say that we have found a compressed form of GNN together with an analytic expression that approximates the relation employed by a GNN that was trained to infer Ω_m from DM halo catalogs. Investigations related to the GNN model by itself proved to be as robust as the model found by Reference (52) for DM halos from N-body simulations for 6 different simulations (namely, PKDGRAD, ABACUS, ENZO, RAMSES, GADGET, and CUBEP3M). We also found that the GNN fails while extrapolating their Ω_m predictions for the galaxy catalogs, but this was expected due to the complex relationship from the astrophysical processes, which are not taken into account in the N-body process. While the GNN failed, the analytical expressions performed good predictions to the galaxy catalogs, after tuning a single free parameter associated with the normalization of the galaxy velocities. Also, the expressions found by SR presented robust results across the different N-body simulations, considering the DM halos and the same normalization factor. We thus reinforce (in agreement with the findings of the work presented in Section 6.1 and in Reference (71)) the fact that the ML suite is learning to make inferences based about Ω_m on more than just spatial correlations, but also on physical relations which are manifested in the phase-space distributions of halos and galaxies.

6.3 The impact of systematic effects

We have seen throughout this thesis that a powerful way to constrain cosmological parameters from galaxy catalogs is to train GNNs to perform field-level likelihood-free inference without imposing cuts on scale. In particular, in Reference (71) (see Section 6.1) we developed models that could accurately infer the value of Ω_m from catalogs that only contain the positions and radial velocities of galaxies, which are robust to astrophysics and subgrid physical models. However, observations are affected by many other real-life effects, including 1) masking, 2) uncertainties in peculiar velocities and radial distances, and 3) different galaxy selections.

Moreover, observations only allow us to measure redshift, intertwining galaxies' radial positions and velocities.

In this section, which was based on Reference (74), we train and test our models on galaxy catalogs, created from different codes from the CAMELS project, that incorporate these observational effects. As we will show, although the presence of these effects degrades the precision and accuracy of the models, increasing the fraction of catalogs where the model breaks down, the fraction of galaxy catalogs where the model performs well is over 90%, demonstrating the potential of these models to constrain cosmological parameters even when applied to real data.

We start by presenting, in Section 6.3.1, the way we incorporate each one of these observational effects into the data set of galaxy catalogs, and how we process these catalogs into graphs. In Section 6.3.2 we briefly discuss the methodology for this work, which is essentially the same as the one employed in Section 6.1. Then, in Section 6.3.3 we present a summary of the results achieved in this work, followed by a discussion in Section 6.3.4.

6.3.1 Data

The data set we use in this work are galaxy catalogs from the CAMELS suite, namely Astrid, SIMBA, IllustrisTNG, SB28, and SWIFT-EAGLE, exactly as presented in Section 6.1.1. One interesting difference is related to the SB28 catalogs, for which in this work we had access to $2,048$ catalogs, exploring more the 28D space of cosmological and astrophysical parameters.

Apart from the similarities, the main difference of this approach are the inclusion of *observational effects* in the galaxy catalogs – which can be regarded as potential systematic effects. In this section we describe the different systematics that we consider, and how we simulate them.

- **Masking.** In real surveys, some fraction of the galaxies are masked out due to a variety of reasons: bright stars, cosmic rays, bad pixels, etc. When we train a ML suite considering the entire sample of objects in the simulations, we are capturing all the information from the cosmic web. In contrast, if we deploy that machinery on masked catalogs, some of that information is erased, and we may end up with less trustworthy predictions. Here we simulate the effect of masks by randomly removing some percentage of the galaxies in the catalog (a different approximation to mask effects, if compared to what we have done in the work of Chapter 2.3.5.1 and in Reference (40)). Since the fraction of survey areas which end up being masked out is typically below 10% of the footprint (266, 360, 361), we simulated masks that eliminate 5% and 10% of the galaxies in our samples.
- **Peculiar velocity uncertainties.** Peculiar velocity cannot be precisely measured, since observations are unable to distinguish between radial (line-of-sight) positions and radial velocities (346, 347, 362–364). In our previous work (71), we built a robust model on the

basis of exact values for the 3D positions and velocities of all the galaxies. However, this phase space information can become blurred or even biased if those positions and velocities are affected by measurement errors, which could then lead to inaccurate (or even biased) predictions. We simulate this effect by adding a random error to the line-of-sight peculiar velocity of each galaxy, v_z, in a catalog. This error is added in two different ways:

  - **Absolute error:**

$$v_z \rightarrow v_z + \mathcal{N}(\mu, \sigma), \tag{6.27}$$

    where $\mathcal{N}(\mu, \sigma)$ is a Gaussian distribution with mean μ and standard deviation σ. We use $\mu = 0$, and both a low ($\sigma = 100$ km/s) as well as a large ($\sigma = 150$ km/s) uncertainty in the velocities. The magnitude of these velocity errors are commensurate with model observational uncertainties related to, e.g., the separation of the peculiar velocities from the Hubble flow, considering the typical velocity dispersion of galaxies inside groups and clusters, which are of order $\sim [300, 500]$ km$/s$ (365).

  - **Relative error:**

$$v_z \rightarrow v_z \left[1 + P\mathcal{N}\left(0, 1\right)\right], \tag{6.28}$$

    where we consider $P = 0.15$ and $P = 0.25$, representing relative errors on the peculiar velocities of 15% and 25%, respectively. The idea is that these velocity errors could come from uncertainties in the redshifts of the galaxies. For a galaxy with peculiar velocity of 200 km$/s$, this amounts to $\sim [30, 50]$ km$/s$, which is comparable to the intrinsic error of spectroscopic surveys such as DESI for galaxies and quasars (366).

- **Line-of-sight distance uncertainties.** In galaxy redshift surveys the radial components of the peculiar velocities are degenerate with the radial positions of the galaxies. We account for this observational constraint by removing the line-of-sight component from the position vector –i.e., we project the galaxies onto a 2D plane. The main modification brought about by this particular test is that, compared with our previous work (71), the graphs are now in 2 dimensions for the edge indices. More details are given in Section 6.3.1.1.

- **Galaxy selection.** In real surveys, galaxies are selected according to some criteria: e.g., objects brighter than some threshold, and/or those whose colors lie within certain ranges. Moreover, some of these criteria are correlated with the clustering and environmental properties (e.g. galaxy colors are related to their positions inside the halos). By folding these selection effects into the training and testing processes we are able to estimate their impact on our predictions for parameters such as Ω_m. We simulate galaxy selection in our catalogs by means of two different criteria:

- **Color**. Since some of the hydrodynamical simulations in CAMELS do not contain galaxy magnitudes (these properties are only available for the IllustrisTNG suite), we employ a selection based on the "quenched" and "not-quenched" galaxies[12]. This definition derives from the values of the specific star formation rate (sSFR = SFR/$\mathrm{M}_\star$ $[\mathrm{yr}^{-1}\mathrm{M}_\odot]$, where a galaxy's SFR is defined as the sum of the individual SFR of all gas cells in its subhalo), according to Reference (155), i.e.
  - **Blue:** $sSFR < 10^{-10.8}\,\mathrm{yr}^{-1}\,\mathrm{M}_\odot$,
  - **Red:** $sSFR > 10^{-10.8}\,\mathrm{yr}^{-1}\,\mathrm{M}_\odot$.
- **Star formation rate**. The second criterion we use is based on the galaxy's SFR, where we define:
  - **Star forming:** $SFR > 0$,
  - **Non star-forming:** $SFR = 0$.

In the present section we show the results that less impact the predictions for each of the above considerations (e.g., for the mask effect we show the results related to masking 5% of the galaxies), otherwise we present both of them. The complete analysis can be seen in Reference (74).

6.3.1.1 Galaxy graphs

Table 10 – **Values of the linking radius found by Optuna for the selected models.**

Model	$\mathbf{r_{link}}$
Relative error: $P = 15\%$	$0.913h^{-1}$ Mpc
Color: Blues	$1.065h^{-1}$ Mpc
Color: Reds	$1.025h^{-1}$ Mpc
Non star-forming	$1.160h^{-1}$ Mpc
Forming stars	$1.231h^{-1}$ Mpc
2D positions and 1D velocity	$1.938h^{-1}$ Mpc

In all the above scenarios we are modifying the CAMELS catalogs in order to include the systematic effects. Thus, we include them in all training, validation, and testing catalogs, while converting them, according to the same prescription of Section 6.1.1.3, to graphs. The main difference relies in the linking radius, which when it was not $\sim 1.25\mathrm{Mpc}/h$, we present according to the effect analyzed in Table 10. This parameter was found with the help of Optuna (243). We emphasize that the above effects do not represent all possible systematic effects that appear in real surveys (369).

[12] We have checked that this choice is similar to the color bi-modality, following Reference (367) for IllustrisTNG CV boxes. For a complete correspondence on IllustrisTNG color and SFR selection, see Reference (368).

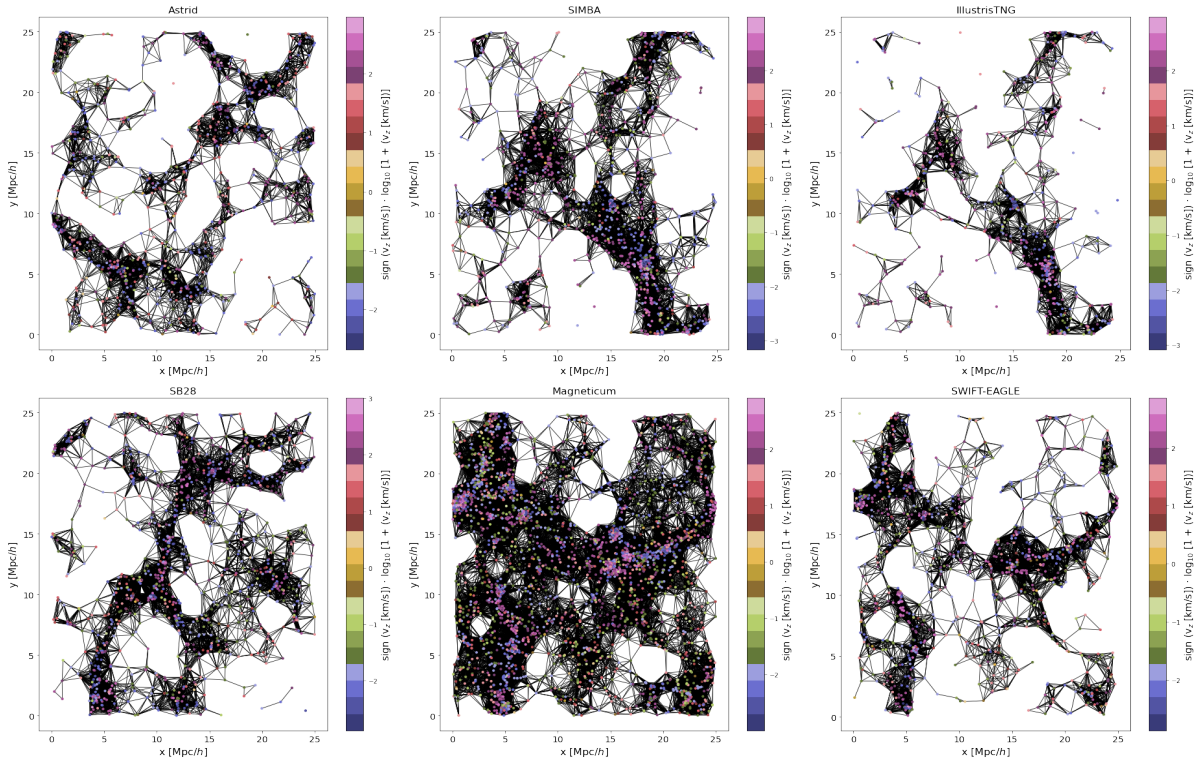

Figure 62 – **Examples of 2D graphs built from galaxy catalogs from different CAMELS simulations: Astrid, SIMBA, IllustrisTNG, Magneticum, SB28, and SWIFT-EAGLE.** The nodes represent the galaxies and their colors correspond to the normalized z component of their peculiar velocity. Galaxies are connected by edges (shown as black lines) if their distance is smaller than the linking radius. We stress that, in this pictorial representation, there are no galaxies which are linked due to PBC. *Source:* Reference (74).

An interesting systematic tested was the *line-of-sight distance uncertainties*. In order to perform this change in the catalogs we consider all the galaxies in the same z plane, as the 2D graphs, show in Figure 62. We present one graph for 1 CV box of Astrid, SIMBA, IllustrisTNG, and Magneticum, and 1 SB/LH box of SB28 and SWIFT-EAGLE. Following the same prescription of Reference (71), we are representing the galaxies by points, colored according to their values of the transformed velocities, and the edge connections by black lines (see Figure 45). Notice that we are not representing the galaxies connected by PBC. Also, we are always selecting galaxies more massive than our minimum stellar mass cut of $M_\star = 1.95 \cdot 10^8$ $M_\odot/h$. By removing one spatial component, this figure shows some important differences compared to the 3D graphs. First, due the fact that we have artificially created galaxies closer to each other (because we removed one spatial component), we allowed more connections (roughly speaking, we have ~ 50 connections per galaxy, 5 times more than in the usual 3D case of Reference (71)). This number is still larger because, as already mentioned in the previous paragraph, the preferred value for the linking radius is larger than in the 3D graphs. This can be indicative of how the GNNs gather information to provide predictions which are still good, even if they do not capture the information from larger scales.

6.3.2 Methodology

We have used the same basic GNN, along with the MNNs architecture for likelihood-free inference, training procedure and optimization, as well as metrics (restricting only to the relative error and reduced chi squared here), that were used to evaluate the models in Sections 6.1.2, 6.1.3, 6.1.4, and 6.1.5. The main difference is the inclusion of different systematics in the galaxy catalogs.

6.3.3 Results

In this section we present the main results of testing our GNN models on galaxy catalogs with different cosmologies, astrophysical parameters, and subgrid physical models from the catalogs used for training, considering the different systematics. For a complete analysis, see Reference (74).

Figure 63 shows the average over the predictions in the LH test set of Astrid, SIMBA, IllustrisTNG, SB28, Magneticum, and SWIFT-EAGLE for the different systematics (namely, masking 5% of the galaxies, relative velocity perturbation of 15%, absolute velocity perturbation of 100km/s, 2D positions and 1D velocity, blue and red galaxies, star-forming and non star-forming galaxies) and comparing it to the results without systematics (see "considering all the galaxies"). This representation is made just to facilitate the discussion (see a complete presentation of all the systematics, as well as a complete representation of the predictions in Reference (74)). Also we present (next to the name of the subgrid physical model, in the legend) the number of catalogs considered with predictions with $\chi^2 < 10$. Catalogs with predictions beyond this threshold are considered as outliers.

The systematic analysis considering 5% of masking, velocity uncertainties, line-of-sight uncertainties, blue, and star-forming galaxies, shows strong evidence that the network is still robust across the different subgrid physical models. This is because all the values for ϵ are close to 12, and χ^2 are around 1, which is in agreement with the results obtained while disregarding those systematics (see, e.g., the results of "considering all the galaxies").

In the case of velocity perturbation, the relative perturbation of 15% (see Equation 6.28) leads to larger error bars, as well as worse scores ($\epsilon > 12\%$), when compared with the absolute (additive) perturbation of 100km/s. A remarkable prediction is that provided by the results using only line-of-sight information – i.e., 2D positions and 1D velocities. Even considering all the galaxies in the same plane, we still get a robust model (with prediction slightly worse for SWIFT-EAGLE), with all the metrics comparable to the model without any systematics. While considering color selection, the predictions show that the network extrapolate better for blue than for red galaxies, a fact which is directly seen in the number of catalogs considered with $\chi^2 > 10$, which are discarded: $6\% \rightarrow 12\%$, from blue to red galaxies. A worse performance is observed for the predictions from star-forming and non star-forming galaxies, where the latter shows the poorest results. In terms of metrics, we have $\epsilon \sim 16\%$ and $\chi^2 \sim 2.4$ for SIMBA for

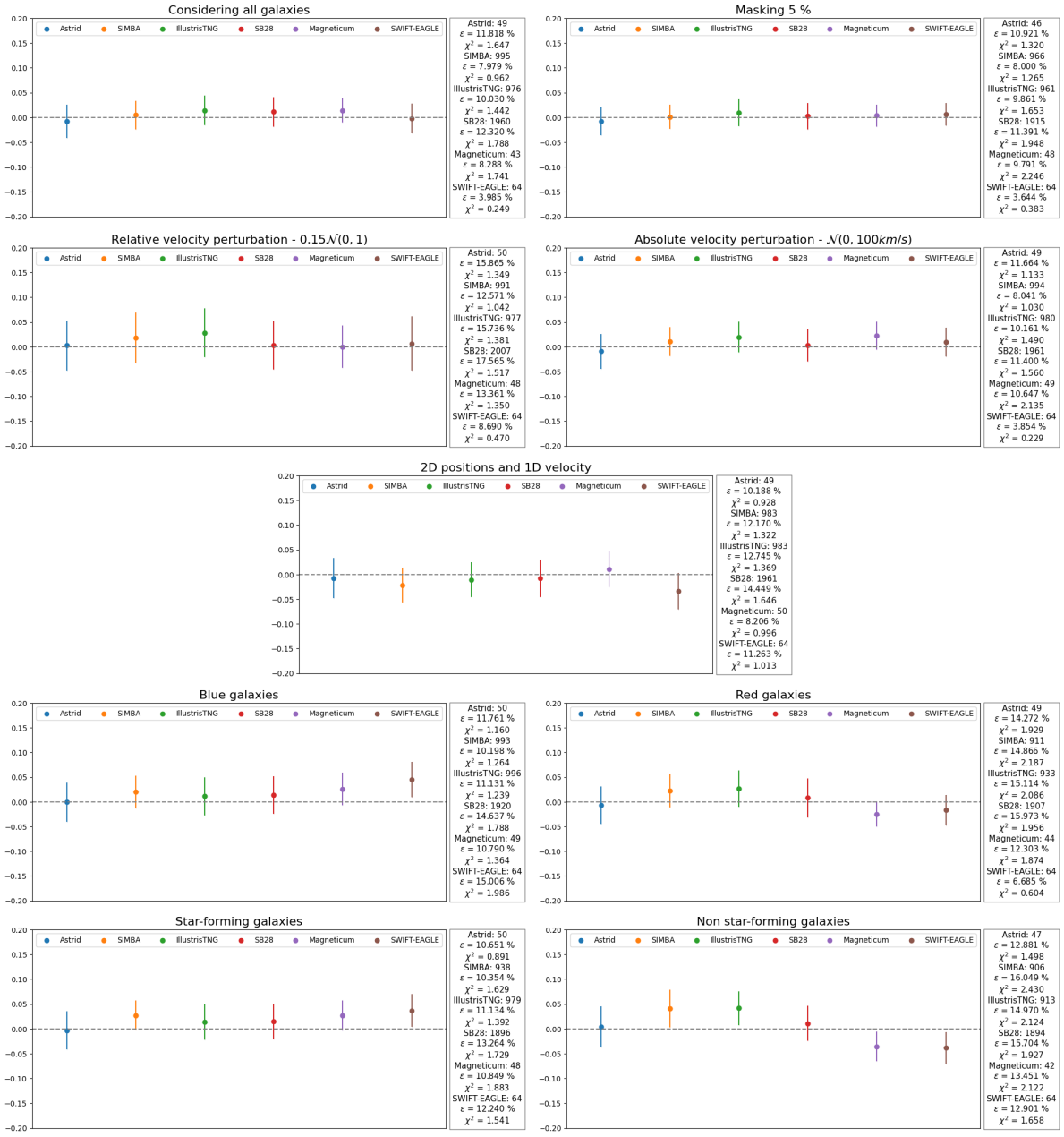

Figure 63 – **Averaged predictions over Astrid, SIMBA, IllustrisTNG, SB28, Magneticum, and SWIFT-EAGLE for the different observational effects compared to the case of disregarding systematics.** We present truth - inference averaged predictions, from the top to bottom, for: on the first row - all the galaxies (without systematics, on the left panel), masking (on the right panel); on the second row - relative velocity perturbation (15%, on the left panel), absolute velocity perturbation (100 km/s, on the right panel); on the third row - line-of-sight distance uncertainties (2D positions and 1D velocities); on the fourth row - blue (on the left panel) and red (on the right panel); on the last row - star-forming (on the left panel) and non star-forming (on the right panel). In the legends we present the metrics for each simulation, followed by the number of boxes with predictions accounting for $\chi^2 < 10$. *Source:* Reference (74).

non star-forming predictions.

6.3.4 Discussion and conclusions

There are many different aspects related to the different performances, across the different systematics considered in this work. When masking the galaxies, we are randomly removing objects from the cosmic-web. This was shown to work for a small fraction of the galaxy content (of course, the results get slightly worse as we enlarge the masked fraction – see Reference (74)). The velocity uncertainties evidentiate the impact over the velocity component of the phase-space information, which leads to a worse performance on the robustness, in both cases of the velocity errors. The main reason for the slightly worse performance when we put all galaxies on a plane can be related to the lesser amount of information. In particular, by doing so we lose information from large scales in the z dimension. The worse performance for color-selected red galaxies implies that the GNN is being impacted by the lower number density and higher clustering information associated with those objects. In the case of star-forming selection, considering non star-forming galaxies means that we are taking into account catalogs with a lower number density of objects. However, those are likely numerical artifacts and, therefore, cannot be trusted (368).

We stress the importance of further investigating the removal of certain galaxy catalogs labeled as "outliers" based on their χ^2 values. Such removal may stem from correlations with specific astrophysical parameters and associated systematic effects, or even from flaws in the definition of galaxy selection criteria, such as color selection, within the LH/SB sets, where variations in parameters may alter color bimodality. Analyzing the latent space and employing Domain Adaptation techniques (see References (370, 371)) could offer avenues for future exploration. However, it is worth noting that this approach has been previously employed in other studies (see References (71, 72, 372)), and the identified outliers constitute only a small fraction of our the test sets. Therefore, these considerations guided our decision to present this solution here.

In conclusion, we have shown that the method proposed by us in Reference (71) (see Section 6.1), to recover cosmological parameters from galaxies, and further developed here, is relatively robust to observational effects. For some of those real-world effects, the results are still very good, while for others there is a larger impact on the accuracy of the recovered parameters. We believe that further improvements can be made by, e.g., training on an even wider parameter space that includes not only cosmology and astrophysics, but also systematic effects. Moreover, we can also design models which are more accurate within a given range of scales and with specific selection criteria. This research represents an important first step towards applying these methods to real galaxy catalogs.

7 DISCUSSION AND CONCLUSIONS

This thesis represents an innovative collection of ML methods for extracting cosmological information. This is accomplished in two ways: first, by obtaining cosmological parameters through traditional pipelines, by improving cosmological covariance matrices, and by performing field-level likelihood-free inferences; and second, by obtaining galaxy properties based on their DM host halo information. After discussing the necessary background of cosmological and ML methods, covered in Chapters 2 and 3, respectively, the achievements related to the problems addressed by our doctoral research were presented in Chapters 4, 5, and 6, dedicated respectively to the improvement of cosmological covariance matrices, halo-galaxy connection, and simulation-based inference. In this final chapter we present a discussion, ideas for improvements, and the main conclusions that can drawn from the entire body of work.

In the Cosmology background, we provide an overview of the foundation concepts in Cosmology. We began in Section 2.1 introducing classical Cosmology, which encompasses Einstein's equations, the FLRW metric, and the Friedmann equations. These serve as the basis for introducing cosmological parameters and discussing the eras of the Universe. In Section 2.2, we briefly described the history of the Universe based on these eras. Section 2.3 introduced various cosmological tools, including correlation functions, power spectra, halo and galaxy bias, and an overview of the halo model. We also touched on the estimation of cosmological parameters using Bayesian theory and covariance matrices, marking the traditional approach to this task (see Section 2.3.5). Linear and nonlinear Cosmology were presented in Sections 2.4 and 2.5, respectively, as means to explain linear and nonlinear scales for the power spectrum, which serves as the primary summary statistic studied throughout the thesis. In particular, in the case of nonlinear Cosmology, we used this opportunity to introduce the main set of simulations utilized in this thesis: N-body and hydrodynamical simulations.

In the ML background chapter, we delved into various ML algorithms to provide a comprehensive understanding of their application. Firstly, we offered a brief review of fundamental ML concepts in Section 3.1. Secondly, in Section 3.3.1, were explored NNs within the context of tabular data. Thirdly, the discussion shifted to CNNs in Section 3.3.2, motivated by their application to matrix data. Fourthly, Section 3.3.3 focused on image denoising techniques, predominantly utilizing CNN blocks. Fifthly, we introduced GNNs and their components, along with the graph data structure, in Section 3.3.4. Sixthly, we presented ideas about SR in Section 3.2.3. Seventhly, tree methods were discussed in Section 3.2.2. Eightly, the k-Nearest Neighbors algorithm for clusterization was introduced in Section 3.2.1. Ninthly, Section 3.5 covered various ML techniques necessary for handling challenges such as imbalanced data sets, stacking different ML predictions, and optimizing ML hyperparameters.

Chapter 4 presents a study focused on developing an efficient method for computing

accurate cosmological covariance matrices, which are essential for parameter inference in Cosmology. By leveraging CNNs for denoising, the method effectively removes noise from input matrices while preserving essential features. We trained our method on covariance matrices computed from a data set comprised of ExSHalos mock catalogs, and showed that the CNN demonstrates robust generalization when applied to covariance matrices from N-body simulations like Quijote. Validation tests confirm the method's efficacy, including MSE reduction, comparison of eigenvalues, and analysis of diagonal values (see Section 4.5). The study also compares the method with mathematical models (presented in Section 4.5.5), revealing its ability to virtually augment sample size. Parameter estimation, presented in Section 4.5.6, further validates the approach, showing significant improvements in parameter constraints. The primary limitation of the model arises from the computational cost associated with running the ExSHalos simulations, albeit still less comparable to running hundreds of N-body simulations – but without this method, one would need to run many thousands of N-body simulations in order to arrive at the same level of accuracy.

Some next steps for this project could include: (i) testing the machinery in matrices from different cosmologies, other than the fiducial ones used in the training stage, in order to assess its generalization capability; (ii) applying the ML suite to more complex and realistic covariance matrices, such as those in redshift space, with multiple tracers of the large-scale structure, and higher-order statistics; (iii) employing this methodology to a real survey, training it on their mocks (accounting for their systematic), and obtaining an accurate and precise covariance matrix from a small number of N-body realizations for it, finally estimating the parameters – and by doing so, reducing the bias in those parameters. Overall, this work underscores the potential of image denoising techniques to enhance cosmological parameter estimation.

Chapter 5 delves into the intricate halo-galaxy connection by employing various ML approaches to predict galaxy properties, including stellar mass, sSFR, radius, and color, based on known DM halo attributes (halo mass, age, spin, concentration, and overdensity). In the first approach (see Section 5.3), we tested different ML algorithms (ERTs, kNN, LGBM, and NN), a combined version of them (stacking their predictions), as well as data augmentation techniques. Validation tests using the MSE and PCC scores have confirmed the method's improvements over other works in the literature, as detailed in Section 5.3.2. Although neither the separate predictions, nor the stacked model, were able to predict very well the full distribution of galaxy properties, the stacked version proved to aggregate the best parts of the separate predictions while mitigating their weaknesses. On the other hand, by leveraging models trained with augmentated data sets we were able to improved the prediction accuracy for the tails of the distribution of galaxy properties, which turns out to be particularly important for sSFR and color – even if the overall distribution remained imperfect.

To address the stochasticity behind the galaxy properties (see Section 5.4), the second approach aimed at obtaining PDFs, instead of single-value (deterministic) predictions. The

excellent performance of this technique resulted in a successful recovery of the complete distribution of galaxy properties, including joint galaxy-galaxy and galaxy-halo distributions, as shown in Section 5.4.2. However, a comparison test between sampling over probabilities and maximum likelihood methods (Section 5.4.2.3) yielded similar results, without necessarily maintaining the overall distribution accurately.

Further investigation should be devoted to: (i) testing other means to improve the overall distribution for maximum likelihood estimation methods (e.g., using different ML methods or exploring other halo-galaxy properties); (ii) employing different probability estimator methods to handle higher-dimensional spaces for the predictions; (iii) conducting a concrete comparison between these two approaches using specific metrics; (iv) including predictions for satellite galaxies, as well as solving the inverse problem (predicting halo properties from the observable galaxies). In that context, Chapter 5 presents two possible solutions to deal with the scatter and the intricate relations of the halo-galaxy connection.

Chapter 6 presents a proof of concept for the use of GNNs in Cosmology, demonstrating novel achievements. Firstly, in Section 6.1, a GNN trained with a MNN on a broader data set of galaxies (specifically, Astrid) is showcased. This robust model can extrapolate predictions to constrain Ω_m across various astrophysical parameters, six different subgrid physical models, and different halo/subhalo finders. Secondly, in Section 6.2, SR was used to translate GNNs into equations that encapsulate the halo/galaxy phase-space information and geometry. While GNNs trained on halos exhibit robustness tested on halo catalogs of many N-body and hydrodynamical simulations, they cannot extrapolate to galaxies; however, the symbolic equation can, after parameter adjustment. Thirdly, in Section 6.3, the effect of observational effects on estimations made by GNNs+MNNs is estimated, showing robustness even when considering these new challenges. Estimations on galaxy catalogs still face many challenges, such accounting for super-sample covariance effects, and σ_8 inference remains elusive.

Additional next steps for this project include: (i) investigating more mechanisms of robustness, such as Domain Adaptation methods; (ii) addressing computational memory issues to handle larger graphs, capable of accommodating more galaxies; (iii) incorporating full probability estimators to replace the MNNs and comparing results with other ML and traditional methods; (iv) marginalizing over observational effects, not only differences in cosmological and astrophysical parameters; (v) investigating the most important physical scales behind the success of these inferences; and (vi) looking for an explanation for the "outliers". Therefore, Chapter 6 presents an alternative ML method for inferring cosmological parameters that does not necessitate cuts on scale, can be interpretable into analytical equations, and which is robust to several known sources of systematic effects.

In conclusion, this thesis has explored innovative applications of ML techniques in Cosmology, addressing key challenges in parameter estimation and exploring in new ways our understanding the halo-galaxy connection. From improving cosmological covariance matrices

using CNNs, to predicting galaxy properties with advanced ML algorithms, each chapter has contributed to advancing our understanding of how to extract valuable information from cosmological data sets. Moreover, the integration of GNNs, MNNs, and SR in Chapter 6 represents a significant leap forward, offering a promising alternative for cosmological parameter inference. By leveraging the power of ML, this work opens new avenues for cosmological research, paving the way for more accurate and efficient methods in the quest to unravel the mysteries of the Cosmos.

REFERENCES

1 Peebles, P. J. E. **Principles of Physical Cosmology**. [*S.l.: s.n.*], 1993.

2 PEACOCK, J. **Cosmological Physics**. Cambridge University Press, 1999. (Cambridge Astrophysics). ISBN 9780521422703. Available at: https://books.google.com.br/books?id=t8O-yylU0j0C.

3 DODELSON, S.; DODELSON, o.; 1941-1969)., A. P. L. . **Modern Cosmology**. Elsevier Science, 2003. ISBN 9780122191411. Available at: https://books.google.com.br/books?id=iBc9TmNLD7kC.

4 SERJEANT, S. **Observational Cosmology**. Cambridge University Press, 2010. ISBN 9780521157155. Available at: https://books.google.com.br/books?id=LfaZRAAACAAJ.

5 MARTIN, J. K.; KOX, A. J.; SCHULMAN, R. **The Collected Papers of Albert Einstein**: Volume 6 - the berlin years: Writings 1914 - 1917. New Jersey: Princeton University Press, 1996. v. 6.

6 FRIEDMAN, A. Uber die krummung des raumes. **Zeitschrift fur Physik**, v. 10, Dec 1922. Available at: https://doi.org/10.1007/BF01332580.

7 Robertson, H. P. Kinematics and World-Structure. **APJ**, v. 82, p. 284, nov. 1935.

8 LEMAITRE, A. G. A Homogeneous Universe of Constant Mass and Increasing Radius accounting for the Radial Velocity of Extra-galactic Nebulae. **Monthly Notices of the Royal Astronomical Society**, v. 91, n. 5, p. 483–490, 03 1931. ISSN 0035-8711. Available at: https://doi.org/10.1093/mnras/91.5.483.

9 WALKER, A. G. On milne's theory of world-structure*. **Proceedings of the London Mathematical Society**, s2-42, n. 1, p. 90–127, 1937. Available at: https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/plms/s2-42.1.90.

10 Lemaître, G. Un Univers homogène de masse constante et de rayon croissant rendant compte de la vitesse radiale des nébuleuses extra-galactiques. **Annales de la Société Scientifique de Bruxelles**, v. 47, p. 49–59, jan. 1927.

11 Hubble, E. A Relation between Distance and Radial Velocity among Extra-Galactic Nebulae. **Proceedings of the National Academy of Science**, v. 15, n. 3, p. 168–173, mar. 1929.

12 Alpher, R. A.; Herman, R. Evolution of the Universe. **NAT**, v. 162, n. 4124, p. 774–775, nov. 1948.

13 Penzias, A. A.; Wilson, R. W. Measurement of the Flux Density of CAS a at 4080 Mc/s. **ApJ**, v. 142, p. 1149, out. 1965.

14 Planck Collaboration *et al.* Planck 2018 results. VI. Cosmological parameters. **AAP**, v. 641, p. A6, set. 2020.

15 Bertone, G.; Hooper, D. History of dark matter. **Reviews of Modern Physics**, v. 90, n. 4, p. 045002, out. 2018.

16 Corbelli, E.; Salucci, P. The extended rotation curve and the dark matter halo of M33. **MNRAS**, v. 311, n. 2, p. 441–447, jan. 2000.

17 Abbott, T. M. C. *et al.* First Cosmology Results using Type Ia Supernovae from the Dark Energy Survey: Constraints on Cosmological Parameters. **Apjl**, v. 872, n. 2, p. L30, fev. 2019.

18 Riess, A. G. *et al.* Observational Evidence from Supernovae for an Accelerating Universe and a Cosmological Constant. **Aj**, v. 116, n. 3, p. 1009–1038, set. 1998.

19 Riess, A. G. *et al.* A Comprehensive Measurement of the Local Value of the Hubble Constant with 1 km s^{-1} Mpc^{-1} Uncertainty from the Hubble Space Telescope and the SH0ES Team. **Apjl**, v. 934, n. 1, p. L7, jul. 2022.

20 Di Valentino, E. *et al.* Cosmology Intertwined III: fσ_8 and S_8. **Astroparticle Physics**, v. 131, p. 102604, set. 2021.

21 Planck Collaboration *et al.* Planck 2013 results. XXII. Constraints on inflation. **AAP**, v. 571, p. A22, nov. 2014.

22 DESI Collaboration *et al.* The DESI Experiment Part I: Science,Targeting, and Survey Design. **arXiv e-prints**, p. arXiv:1611.00036, out. 2016.

23 Laureijs, R. *et al.* Euclid Definition Study Report. **arXiv e-prints**, p. arXiv:1110.3193, out. 2011.

24 Amendola, L. *et al.* Cosmology and Fundamental Physics with the Euclid Satellite. **Living Reviews in Relativity**, v. 16, n. 1, p. 6, set. 2013.

25 Racca, G. D. *et al.* The Euclid mission design. *In*: MacEwen, H. A. *et al.* (ed.). **Space Telescopes and Instrumentation 2016: Optical, Infrared, and Millimeter Wave**. [*S.l.: s.n.*], 2016. (Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, v. 9904), p. 99040O.

26 Euclid Collaboration: Castro, T. *et al.* Euclid preparation. XXIV. Calibration of the halo mass function in $\Lambda(\nu)$CDM cosmologies. **arXiv e-prints**, p. arXiv:2208.02174, ago. 2022.

27 Takada, M. *et al.* Extragalactic science, cosmology, and Galactic archaeology with the Subaru Prime Focus Spectrograph. **PAsJ**, v. 66, n. 1, p. R1, fev. 2014.

28 Benitez, N. *et al.* J-PAS: The Javalambre-Physics of the Accelerated Universe Astrophysical Survey. **arXiv e-prints**, p. arXiv:1403.5237, mar. 2014.

29 Taylor, A.; Braun, R. (ed.). **Science with the Square Kilometer Array : a next generation world radio observatory**. [*S.l.: s.n.*], 1999.

30 Spergel, D. *et al.* Wide-Field InfrarRed Survey Telescope-Astrophysics Focused Telescope Assets WFIRST-AFTA 2015 Report. **arXiv e-prints**, p. arXiv:1503.03757, mar. 2015.

31 Pontoppidan, K. M. *et al.* The JWST Early Release Observations. **ApJL**, v. 936, n. 1, p. L14, set. 2022.

32 Hahn, C. *et al.* Constraining M_ν with the bispectrum. Part I. Breaking parameter degeneracies. **JCAP**, v. 2020, n. 3, p. 040, mar. 2020.

33 Uhlemann, C. *et al.* Fisher for complements: extracting cosmology and neutrino mass from the counts-in-cells PDF. **MNRAS**, v. 495, n. 4, p. 4006–4027, jul. 2020.

34 Gualdi, D.; Gil-Marín, H.; Verde, L. Joint analysis of anisotropic power spectrum, bispectrum and trispectrum: application to N-body simulations. **JCAP**, v. 2021, n. 7, p. 008, jul. 2021.

35 Banerjee, A.; Abel, T. Nearest neighbour distributions: New statistical measures for cosmological clustering. **MNRAS**, v. 500, n. 4, p. 5479–5499, jan. 2021.

36 Efron, B. **The Jackknife, the Bootstrap and other resampling plans**. [*S.l.: s.n.*]: SIAM, 1982.

37 Taylor, A.; Joachimi, B.; Kitching, T. Putting the precision in precision cosmology: How accurate should your data covariance matrix be? **MNRAS**, v. 432, n. 3, p. 1928–1946, jul. 2013.

38 Heavens, A. F. *et al.* Massive data compression for parameter-dependent covariance matrices. **MNRAS**, v. 472, n. 4, p. 4244–4250, dez. 2017.

39 Chartier, N.; Wandelt, B. D. CARPool covariance: fast, unbiased covariance estimation for large-scale structure observables. **MNRAS**, v. 509, n. 2, p. 2220–2233, jan. 2022.

40 de Santi, N. S. M.; Abramo, L. R. Improving cosmological covariance matrices with machine learning. **JCAP**, v. 2022, n. 9, p. 013, set. 2022.

41 IVEZIĆ, Ž. *et al.* **Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data**. Princeton University Press, 2014. (Princeton Series in Modern Observational Astronomy). ISBN 9780691151687. Available at: https://books.google.com.br/books?id=h2eYDwAAQBAJ.

42 Perez, L. A. *et al.* Constraining cosmology with machine learning and galaxy clustering: the CAMELS-SAM suite. **arXiv e-prints**, p. arXiv:2204.02408, abr. 2022.

43 Cranmer, K.; Brehmer, J.; Louppe, G. The frontier of simulation-based inference. **Proceedings of the National Academy of Science**, v. 117, n. 48, p. 30055–30062, dez. 2020.

44 Ravanbakhsh, S. *et al.* Estimating Cosmological Parameters from the Dark Matter Distribution. **arXiv e-prints**, p. arXiv:1711.02033, nov. 2017.

45 Ntampaka, M. *et al.* A Hybrid Deep Learning Approach to Cosmological Constraints from Galaxy Redshift Surveys. **ApJ**, v. 889, n. 2, p. 151, fev. 2020.

46 Mangena, T.; Hassan, S.; Santos, M. G. Constraining the reionization history using deep learning from 21-cm tomography with the Square Kilometre Array. **MNRAS**, v. 494, n. 1, p. 600–606, maio 2020.

47 Hassan, S.; Andrianomena, S.; Doughty, C. Constraining the astrophysics and cosmology from 21 cm tomography using deep learning with the SKA. **MNRAS**, v. 494, n. 4, p. 5761–5774, jun. 2020.

48 Villaescusa-Navarro, F. *et al.* Multifield Cosmology with Artificial Intelligence. **arXiv e-prints**, p. arXiv:2109.09747, set. 2021.

49 Cole, A. *et al.* Fast and credible likelihood-free cosmology with truncated marginal neural ratio estimation. **JCAP**, v. 2022, n. 9, p. 004, set. 2022.

50 Villanueva-Domingo, P.; Villaescusa-Navarro, F. Learning Cosmology and Clustering with Cosmic Graphs. **ApJ**, v. 937, n. 2, p. 115, out. 2022.

51 Makinen, T. L. *et al.* The Cosmic Graph: Optimal Information Extraction from Large-Scale Structure using Catalogues. **arXiv e-prints**, p. arXiv:2207.05202, jul. 2022.

52 Shao, H. *et al.* Robust field-level inference with dark matter halos. **arXiv e-prints**, p. arXiv:2209.06843, set. 2022.

53 Jo, Y.; Kim, J.-h. Machine-assisted semi-simulation model (MSSM): estimating galactic baryonic properties from their dark matter using a machine trained on hydrodynamic simulations. **MNRAS**, v. 489, n. 3, p. 3565–3581, nov. 2019.

54 Yip, J. H. T. *et al.* From Dark Matter to Galaxies with Convolutional Neural Networks. **arXiv e-prints**, p. arXiv:1910.07813, out. 2019.

55 Zhang, X. *et al.* From Dark Matter to Galaxies with Convolutional Networks. **arXiv e-prints**, p. arXiv:1902.05965, fev. 2019.

56 KAMDAR, H. M.; TURK, M. J.; BRUNNER, R. J. Machine learning and cosmological simulations – ii. hydrodynamical simulations. **Monthly Notices of the Royal Astronomical Society**, v. 457, p. 1162–1179, 2016.

57 Wadekar, D. *et al.* Modeling assembly bias with machine learning and symbolic regression. **arXiv e-prints**, p. arXiv:2012.00111, nov. 2020.

58 Kasmanoff, N. *et al.* dm2gal: Mapping Dark Matter to Galaxies with Neural Networks. **arXiv e-prints**, p. arXiv:2012.00186, nov. 2020.

59 Moster, B. P. *et al.* GalaxyNet: connecting galaxies and dark matter haloes with deep neural networks and reinforcement learning in large volumes. **MNRAS**, v. 507, n. 2, p. 2115–2136, out. 2021.

60 McGibbon, R. J.; Khochfar, S. Multi-epoch machine learning 1: Unravelling nature versus nurture for galaxy formation. **MNRAS**, v. 513, n. 4, p. 5423–5437, jul. 2022.

61 Shao, H. *et al.* Finding Universal Relations in Subhalo Properties with Artificial Intelligence. **ApJ**, v. 927, n. 1, p. 85, mar. 2022.

62 von Marttens, R. *et al.* Inferring galaxy dark halo properties from visible matter with machine learning. **MNRAS**, v. 516, n. 3, p. 3924–3943, nov. 2022.

63 Villanueva-Domingo, P. *et al.* Weighing the Milky Way and Andromeda with Artificial Intelligence. **arXiv e-prints**, p. arXiv:2111.14874, nov. 2021.

64 Delgado, A. M. *et al.* Modelling the galaxy-halo connection with machine learning. **MNRAS**, v. 515, n. 2, p. 2733–2746, set. 2022.

65 de Santi, N. S. M. *et al.* Mimicking the halo-galaxy connection using machine learning. **MNRAS**, v. 514, n. 2, p. 2463–2478, ago. 2022.

66 Jespersen, C. K. *et al.* `Mangrove`: Learning Galaxy Properties from Merger Trees. **arXiv e-prints**, p. arXiv:2210.13473, out. 2022.

67 Villanueva-Domingo, P. *et al.* Inferring Halo Masses with Graph Neural Networks. **ApJ**, v. 935, n. 1, p. 30, ago. 2022.

68 Lovell, C. C. *et al.* A machine learning approach to mapping baryons on to dark matter haloes using the EAGLE and C-EAGLE simulations. **MNRAS**, v. 509, n. 4, p. 5046–5061, fev. 2022.

69 Rodrigues, N. V. N. *et al.* High-fidelity reproduction of central galaxy joint distributions with Neural Networks. **arXiv e-prints**, p. arXiv:2301.06398, jan. 2023.

70 Villaescusa-Navarro, F. *et al.* Cosmology with One Galaxy? **ApJ**, v. 929, n. 2, p. 132, abr. 2022.

71 de Santi, N. S. M. *et al.* Robust Field-level Likelihood-free Inference with Galaxies. **ApJ**, v. 952, n. 1, p. 69, jul. 2023.

72 Shao, H. *et al.* A Universal Equation to Predict Ω_m from Halo and Galaxy Catalogs. **ApJ**, v. 956, n. 2, p. 149, out. 2023.

73 Ni, Y. *et al.* The CAMELS project: Expanding the galaxy formation model space with new ASTRID and 28-parameter TNG and SIMBA suites. **arXiv e-prints**, p. arXiv:2304.02096, abr. 2023.

74 de Santi, N. S. M. *et al.* Field-level simulation-based inference with galaxy catalogs: the impact of systematic effects. **arXiv e-prints**, p. arXiv:2310.15234, out. 2023.

75 Mukhanov, V. **Physical Foundations of Cosmology**. [*S.l.: s.n.*], 2005.

76 AMENDOLA, L.; TSUJIKAWA, S. **Dark Energy: Theory and Observations**. Cambridge University Press, 2010. ISBN 9781139488570. Available at: https://books.google.com.br/books?id=Xge0hg_AIIYC.

77 CARROLL, S. M. **Spacetime and Geometry**: An introduction to general relativity. San Francisco: Addison Wesley, 2004.

78 FOSTER, J.; NIGHTINGALE, J. D. **A Short Course in General Relativity**. 3. ed. New York: Springer, 2006.

79 HARTLE, J. B. **Gravity**: An introduction to einstein's general relativity. San Francisco: Addison-Wesley, 2003.

80 WORKMAN, R. L.; OTHERS. Review of Particle Physics. **PTEP**, v. 2022, p. 083C01, 2022.

81 Kosowsky, A.; Milosavljevic, M.; Jimenez, R. Efficient cosmological parameter estimation from microwave background anisotropies. **PRD**, v. 66, n. 6, p. 063007, set. 2002.

82 KAMIONKOWSKI, M.; RIESS, A. G. The hubble tension and early dark energy. **Annual Review of Nuclear and Particle Science**, v. 73, n. 1, p. 153–180, 2023. Available at: https://doi.org/10.1146/annurev-nucl-111422-024107.

83 NASA. **History of the Universe**. 2020. [Online; accessed May 6th, 2024]. Available at: https://science.nasa.gov/resource/history-of-the-universe/?category=universe.

84 Baumann, D. TASI Lectures on Inflation. **arXiv e-prints**, p. arXiv:0907.5424, jul. 2009.

85 Achúcarro, A. *et al.* Inflation: Theory and Observations. **arXiv e-prints**, p. arXiv:2203.08128, mar. 2022.

86 Steigman, G. Primordial Nucleosynthesis in the Precision Cosmology Era. **Annual Review of Nuclear and Particle Science**, v. 57, n. 1, p. 463–491, nov. 2007.

87 Rubakov, V. A.; Gorbunov, D. S. **Introduction to the Theory of the Early Universe: Hot Big Bang Theory**. [*S.l.: s.n.*], 2018.

88 Wong, W. Y.; Moss, A.; Scott, D. How well do we understand cosmological recombination? **MNRAS**, v. 386, n. 2, p. 1023–1028, maio 2008.

89 Bassett, B.; Hlozek, R. Baryon acoustic oscillations. *In*: Ruiz-Lapuente, P. (ed.). **Dark Energy: Observational and Theoretical Approaches**. [*S.l.: s.n.*], 2010. p. 246.

90 Hu, W. CMB temperature and polarization anisotropy fundamentals. **Annals of Physics**, v. 303, n. 1, p. 203–225, jan. 2003.

91 Pritchard, J. R.; Loeb, A. 21 cm cosmology in the 21st century. **Reports on Progress in Physics**, v. 75, n. 8, p. 086901, ago. 2012.

92 Zaroubi, S. The Epoch of Reionization. *In*: Wiklind, T.; Mobasher, B.; Bromm, V. (ed.). **The First Galaxies**. [*S.l.: s.n.*], 2013. (Astrophysics and Space Science Library, v. 396), p. 45.

93 Weinberg, S. **Cosmology**. [*S.l.: s.n.*], 2008.

94 Feldman, H. A.; Kaiser, N.; Peacock, J. A. Power-Spectrum Analysis of Three-dimensional Redshift Surveys. **APJ**, v. 426, p. 23, maio 1994.

95 Bardeen, J. M. *et al.* The Statistics of Peaks of Gaussian Random Fields. **APJ**, v. 304, p. 15, maio 1986.

96 Hand, N. *et al.* nbodykit: An Open-source, Massively Parallel Toolkit for Large-scale Structure. **AJ**, v. 156, n. 4, p. 160, out. 2018.

97 Cooray, A.; Sheth, R. Halo models of large scale structure. **PHYSREP**, v. 372, n. 1, p. 1–129, dez. 2002.

98 Press, W. H.; Schechter, P. Formation of Galaxies and Clusters of Galaxies by Self-Similar Gravitational Condensation. **APJ**, v. 187, p. 425–438, fev. 1974.

99 Lacey, C.; Cole, S. Merger Rates in Hierarchical Models of Galaxy Formation - Part Two - Comparison with N-Body Simulations. **MNRAS**, v. 271, p. 676, dez. 1994.

100 Sheth, R. K.; Tormen, G. Large-scale bias and the peak background split. **MNRAS**, v. 308, n. 1, p. 119–126, set. 1999.

101 Tinker, J. *et al.* Toward a Halo Mass Function for Precision Cosmology: The Limits of Universality. **APJ**, v. 688, n. 2, p. 709–728, dez. 2008.

102 Bhattacharya, S. *et al.* Mass Function Predictions Beyond ΛCDM. **APJ**, v. 732, n. 2, p. 122, maio 2011.

103 Tinker, J. L. *et al.* The Large-scale Bias of Dark Matter Halos: Numerical Calibration and Model Tests. **APJ**, v. 724, n. 2, p. 878–886, dez. 2010.

104 Morrison, C. B.; Schneider, M. D. On estimating cosmology-dependent covariance matrices. **JCAP**, v. 2013, n. 11, p. 009, nov. 2013.

105 Kodwani, D.; Alonso, D.; Ferreira, P. The effect on cosmological parameter estimation of a parameter dependent covariance matrix. **The Open Journal of Astrophysics**, v. 2, n. 1, p. 3, mar. 2019.

106 GUTH, A. H. Inflationary universe: A possible solution to the horizon and flatness problems. **Phys. Rev. D**, American Physical Society, v. 23, p. 347–356, Jan 1981. Available at: https://link.aps.org/doi/10.1103/PhysRevD.23.347.

107 Bardeen, J. M. *et al.* The Statistics of Peaks of Gaussian Random Fields. **ApJ**, v. 304, p. 15, maio 1986.

108 Eisenstein, D. J.; Hu, W. Baryonic Features in the Matter Transfer Function. **ApJ**, v. 496, n. 2, p. 605–614, mar. 1998.

109 HARRISON, E. R. Fluctuations at the threshold of classical cosmology. **Phys. Rev. D**, American Physical Society, v. 1, p. 2726–2730, May 1970. Available at: https://link.aps.org/doi/10.1103/PhysRevD.1.2726.

110 ZELDOVICH, Y. B. A Hypothesis, Unifying the Structure and the Entropy of the Universe. **Monthly Notices of the Royal Astronomical Society**, v. 160, n. 1, p. 1P–3P, 10 1972. ISSN 0035-8711. Available at: https://doi.org/10.1093/mnras/160.1.1P.

111 Lewis, A.; Challinor, A. **CAMB: Code for Anisotropies in the Microwave Background**. 2011. ascl:1102.026 p.

112 Villaescusa-Navarro, F. *et al.* The Quijote Simulations. **APJS**, v. 250, n. 1, p. 2, set. 2020.

113 Crocce, M.; Scoccimarro, R. Renormalized cosmological perturbation theory. **PRD**, v. 73, n. 6, p. 063519, mar. 2006.

114 Vlah, Z.; Seljak, U.; Baldauf, T. Lagrangian perturbation theory at one loop order: Successes, failures, and improvements. **PRD**, v. 91, n. 2, p. 023508, jan. 2015.

115 Cooray, A.; Sheth, R. Halo models of large scale structure. **PhysRep**, v. 372, n. 1, p. 1–129, dez. 2002.

116 White, M. The Zel'dovich approximation. **MNRAS**, v. 439, n. 4, p. 3630–3640, abr. 2014.

117 Osato, K. *et al.* Perturbation theory challenge for cosmological parameters estimation: Matter power spectrum in real space. **PRD**, v. 99, n. 6, p. 063530, mar. 2019.

118 Schmidt, F. Towards a self-consistent halo model for the nonlinear large-scale structure. **PRD**, v. 93, n. 6, p. 063512, mar. 2016.

119 Sellwood, J. A. The art of N-body building. **ARAA**, v. 25, p. 151–186, jan. 1987.

120 Trenti, M.; Hut, P. Gravitational N-body Simulations. **arXiv e-prints**, p. arXiv:0806.3950, jun. 2008.

121 Springel, V. *et al.* Simulations of the formation, evolution and clustering of galaxies and quasars. **NAT**, v. 435, n. 7042, p. 629–636, jun. 2005.

122 Skillman, S. W. *et al.* Dark Sky Simulations: Early Data Release. **arXiv e-prints**, p. arXiv:1407.2600, jul. 2014.

123 Riebe, K. *et al.* The MultiDark Database: Release of the Bolshoi and MultiDark Cosmological Simulations. **arXiv e-prints**, p. arXiv:1109.0003, ago. 2011.

124 Springel, V. The cosmological simulation code GADGET-2. **MNRAS**, v. 364, n. 4, p. 1105–1134, dez. 2005.

125 Vogelsberger, M. *et al.* Cosmological simulations of galaxy formation. **Nature Reviews Physics**, v. 2, n. 1, p. 42–66, jan. 2020.

126 Kravtsov, A. V.; Klypin, A. A.; Khokhlov, A. M. Adaptive Refinement Tree: A New High-Resolution N-Body Code for Cosmological Simulations. **ApJS**, v. 111, n. 1, p. 73–94, jul. 1997.

127 Kravtsov, A. V. **Writing a PM code**. 2002. https://astro.uchicago.edu/~andrey/talks/PM/pm_slides.pdfhttps://astro.uchicago.edu/ãndrey/talks/PM/pm_slides.pdf. Accessed: 2024-05-06.

128 Klypin, A.; Holtzman, J. Particle-Mesh code for cosmological simulations. **arXiv e-prints**, p. astro–ph/9712217, dez. 1997.

129 Hellwing, W. **A short introduction to numerical methods used in cosmological N-body simulations**. 2015. https://www.pta.edu.pl/pliki/proc/kielce15/v2p58.pdf5/v2p58.pdf. Accessed: 2024-05-06.

130 Hahn, O.; Abel, T. Multi-scale initial conditions for cosmological simulations. **MNRAS**, v. 415, n. 3, p. 2101–2121, ago. 2011.

131 Oyaizu, H. Nonlinear evolution of f(R) cosmologies. I. Methodology. **PRD**, v. 78, n. 12, p. 123523, dez. 2008.

132 Knebe, A. *et al.* Haloes gone MAD: The Halo-Finder Comparison Project. **MNRAS**, v. 415, n. 3, p. 2293–2318, ago. 2011.

133 Gómez, J. S. *et al.* Halo merger tree comparison: impact on galaxy formation models. **MNRAS**, v. 510, n. 4, p. 5500–5519, mar. 2022.

134 Springel, V. *et al.* Populating a cluster of galaxies - I. Results at [formmu2]z=0. **MNRAS**, v. 328, n. 3, p. 726–750, dez. 2001.

135 Dolag, K. *et al.* Substructures in hydrodynamical cluster simulations. **MNRAS**, v. 399, n. 2, p. 497–514, out. 2009.

136 Behroozi, P. S.; Wechsler, R. H.; Wu, H.-Y. The ROCKSTAR Phase-space Temporal Halo Finder and the Velocity Offsets of Cluster Cores. **ApJ**, v. 762, n. 2, p. 109, jan. 2013.

137 Elahi, P. J. *et al.* Hunting for galaxies and halos in simulations with velociraptor. **PASA**, v. 36, p. e021, Jan 2019.

138 Cañas, R. *et al.* Introducing a new, robust galaxy-finder algorithm for simulations. **MNRAS**, v. 482, n. 2, p. 2039–2064, Jan 2019.

139 Scoccimarro, R.; Sheth, R. K. PTHALOS: a fast method for generating mock galaxy distributions. **MNRAS**, v. 329, n. 3, p. 629–640, jan. 2002.

140 Chuang, C.-H. *et al.* EZmocks: extending the Zel'dovich approximation to generate mock galaxy catalogues with accurate clustering statistics. **MNRAS**, v. 446, n. 3, p. 2621–2628, jan. 2015.

141 Monaco, P. *et al.* An accurate tool for the fast generation of dark matter halo catalogues. **MNRAS**, v. 433, n. 3, p. 2389–2402, ago. 2013.

142 Kitaura, F. S.; Yepes, G.; Prada, F. Modelling baryon acoustic oscillations with perturbation theory and stochastic halo biasing. **MNRAS**, v. 439, p. L21–L25, mar. 2014.

143 Avila, S. *et al.* HALOGEN: a tool for fast generation of mock halo catalogues. **MNRAS**, v. 450, n. 2, p. 1856–1867, jun. 2015.

144 Agrawal, A. *et al.* Generating log-normal mock catalog of galaxies in redshift space. **JCAP**, v. 2017, n. 10, p. 003, out. 2017.

145 Izard, A.; Fosalba, P.; Crocce, M. ICE-COLA: fast simulations for weak lensing observables. **MNRAS**, v. 473, n. 3, p. 3051–3061, jan. 2018.

146 Voivodic, R.; Lima, M.; Abramo, L. R. Excursion Set Halos – ExSHalos: A New Parameter Free Method for Fast Generation of Halo Catalogues. **arXiv e-prints**, p. arXiv:1906.06630, jun. 2019.

147 Balaguera-Antolínez, A. *et al.* BAM: bias assignment method to generate mock catalogues. **MNRAS**, v. 483, n. 1, p. L58–L63, fev. 2019.

148 Balaguera-Antolínez, A. *et al.* One simulation to have them all: performance of the Bias Assignment Method against N-body simulations. **MNRAS**, v. 491, n. 2, p. 2565–2575, jan. 2020.

149 Blot, L. *et al.* Comparing approximate methods for mock catalogues and covariance matrices II: power spectrum multipoles. **MNRAS**, v. 485, n. 2, p. 2806–2824, maio 2019.

150 Bond, J. R. *et al.* Excursion Set Mass Functions for Hierarchical Gaussian Fluctuations. **ApJ**, v. 379, p. 440, out. 1991.

151 Maggiore, M.; Riotto, A. The Halo Mass Function from Excursion Set Theory. I. Gaussian Fluctuations with Non-Markovian Dependence on the Smoothing Scale. **ApJ**, v. 711, n. 2, p. 907–927, mar. 2010.

152 Matsubara, T. Nonlinear perturbation theory with halo bias and redshift-space distortions via the Lagrangian picture. **PRD**, v. 78, n. 8, p. 083519, out. 2008.

153 Crain, R. A.; van de Voort, F. Hydrodynamical Simulations of the Galaxy Population: Enduring Successes and Outstanding Challenges. **ARAA**, v. 61, p. 473–515, ago. 2023.

154 Bird, S. *et al.* The ASTRID simulation: galaxy formation and reionization. **MNRAS**, v. 512, n. 3, p. 3703–3716, maio 2022.

155 Davé, R. *et al.* SIMBA: Cosmological simulations with black hole growth and feedback. **MNRAS**, v. 486, n. 2, p. 2827–2849, jun. 2019.

156 Pillepich, A. *et al.* Simulating galaxy formation with the IllustrisTNG model. **MNRAS**, v. 473, n. 3, p. 4077–4106, jan. 2018.

157 Hirschmann, M. *et al.* Cosmological simulations of black hole growth: AGN luminosities and downsizing. **MNRAS**, v. 442, n. 3, p. 2304–2324, ago. 2014.

158 Schaye, J. *et al.* The EAGLE project: simulating the evolution and assembly of galaxies and their environments. **MNRAS**, v. 446, n. 1, p. 521–554, jan. 2015.

159 Springel, V. Smoothed Particle Hydrodynamics in Astrophysics. **ARAA**, v. 48, p. 391–430, set. 2010.

160 Smith, B.; Sigurdsson, S.; Abel, T. Metal cooling in simulations of cosmic structure formation. **MNRAS**, v. 385, n. 3, p. 1443–1454, abr. 2008.

161 Wiersma, R. P. C.; Schaye, J.; Smith, B. D. The effect of photoionization on the cooling rates of enriched, astrophysical plasmas. **MNRAS**, v. 393, n. 1, p. 99–107, fev. 2009.

162 Oppenheimer, B. D.; Davé, R. Mass, metal, and energy feedback in cosmological simulations. **MNRAS**, v. 387, n. 2, p. 577–600, jun. 2008.

163 Tornatore, L. *et al.* Chemical enrichment of galaxy clusters from hydrodynamical simulations. **MNRAS**, v. 382, n. 3, p. 1050–1072, dez. 2007.

164 Wiersma, R. P. C. *et al.* Chemical enrichment in cosmological, smoothed particle hydrodynamics simulations. **MNRAS**, v. 399, n. 2, p. 574–600, out. 2009.

165 Schaye, J. *et al.* The EAGLE project: simulating the evolution and assembly of galaxies and their environments. **MNRAS**, v. 446, n. 1, p. 521–554, jan. 2015.

166 Kaviraj, S. *et al.* The Horizon-AGN simulation: evolution of galaxy properties over cosmic time. **MNRAS**, v. 467, n. 4, p. 4739–4752, jun. 2017.

167 Pakmor, R. *et al.* The MillenniumTNG Project: the hydrodynamical full physics simulation and a first look at its galaxy clusters. **MNRAS**, v. 524, n. 2, p. 2539–2555, set. 2023.

168 Davé, R. *et al.* SIMBA: Cosmological simulations with black hole growth and feedback. **MNRAS**, v. 486, n. 2, p. 2827–2849, jun. 2019.

169 Feldmann, R. *et al.* FIREbox: simulating galaxies at high dynamic range in a cosmological volume. **MNRAS**, v. 522, n. 3, p. 3831–3860, jul. 2023.

170 Li, C.; White, S. D. M. The distribution of stellar mass in the low-redshift Universe. **MNRAS**, v. 398, n. 4, p. 2177–2187, out. 2009.

171 Wright, A. H. *et al.* Galaxy And Mass Assembly (GAMA): the galaxy stellar mass function to z = 0.1 from the r-band selected equatorial regions. **MNRAS**, v. 470, n. 1, p. 283–302, set. 2017.

172 WECHSLER, R. H.; TINKER, J. L. The connection between galaxies and their dark matter halos. **Annual Review of Astronomy and Astrophysics**, Annual Reviews, v. 56, n. 1, p. 435–487, Sep 2018. ISSN 1545-4282. Available at: http://dx.doi.org/10.1146/annurev-astro-081817-051756.

173 Somerville, R. S.; Primack, J. R. Semi-analytic modelling of galaxy formation: the local Universe. **MNRAS**, v. 310, n. 4, p. 1087–1110, dez. 1999.

174 Hirschmann, M.; De Lucia, G.; Fontanot, F. Galaxy assembly, stellar feedback and metal enrichment: the view from the GAEA model. **MNRAS**, v. 461, n. 2, p. 1760–1785, set. 2016.

175 Henriques, B. M. B. *et al.* L-GALAXIES 2020: Spatially resolved cold gas phases, star formation, and chemical enrichment in galactic discs. **MNRAS**, v. 491, n. 4, p. 5795–5814, fev. 2020.

176 Contreras, S.; Angulo, R. E.; Zennaro, M. A flexible subhalo abundance matching model for galaxy clustering in redshift space. **MNRAS**, v. 508, n. 1, p. 175–189, nov. 2021.

177 Hearin, A. P. *et al.* Introducing decorated HODs: modelling assembly bias in the galaxy-halo connection. **MNRAS**, v. 460, n. 3, p. 2552–2570, ago. 2016.

178 BISHOP, C. **Neural Networks for Pattern Recognition**. Clarendon Press, 1995. (Advanced Texts in Econometrics). ISBN 9780198538646. Available at: https://books.google.com.br/books?id=T0S0BgAAQBAJ.

179 CHOLLET, F. **Deep Learning with Python**. Manning Publications Company, 2017. ISBN 9781617294433. Available at: https://books.google.com.br/books?id=Yo3CAQAACAAJ.

180 SAMUEL, A. L. Some studies in machine learning using the game of checkers. **IBM Journal of Research and Development**, v. 3, n. 3, p. 210–229, 1959.

181 Analytics Vidhya. **Bias and Variance in Machine Learning**. 2024. [Online; accessed May 6th, 2024]. Available at: https://editor.analyticsvidhya.com/uploads/52858bias-and-variance-in-machine-learning3.png.

182 PEDREGOSA, F. *et al.* Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

183 Xu, X. *et al.* A First Look at Creating Mock Catalogs with Machine Learning Techniques. **ApJ**, v. 772, n. 2, p. 147, ago. 2013.

184 AGARWAL, S.; DAVé, R.; BASSETT, B. A. Painting galaxies into dark matter halos using machine learning. **Monthly Notices of the Royal Astronomical Society**, v. 478, p. 3410–3422, 2018.

185 IBM. **Decision Tree**. 2024. [Online; accessed May 6th, 2024]. Available at: https://www.ibm.com/content/dam/connectedassets-adobe-cms/worldwide-content/cdp/cf/ul/g/df/de/Decision-Tree.png.

186 BREIMAN, L. Random Forests. **Machine Learning**, v. 45, p. 3871–3889, out. 2001.

187 Ball, N. M. *et al.* Robust Machine Learning Applied to Astronomical Data Sets. I. Star-Galaxy Classification of the Sloan Digital Sky Survey DR3 Using Decision Trees. **ApJ**, v. 650, n. 1, p. 497–509, out. 2006.

188 Xu, X. *et al.* Predicting halo occupation and galaxy assembly bias with machine learning. **MNRAS**, v. 507, n. 4, p. 4879–4899, nov. 2021.

189 GEURTS, P.; ERNST, D.; WEHENKEL, L. Extremely randomized trees. **Machine Learning**, v. 63, p. 3–42, 04 2006.

190 Lovell, C. C. *et al.* A machine learning approach to mapping baryons onto dark matter halos using the EAGLE and C-EAGLE simulations. **arXiv e-prints**, p. arXiv:2106.04980, jun. 2021.

191 KE, G. *et al.* Lightgbm: A highly efficient gradient boosting decision tree. *In*: GUYON, I. *et al.* (ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2017. v. 30. Available at: https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

192 FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. **The Annals of Statistics**, Institute of Mathematical Statistics, v. 29, n. 5, p. 1189 – 1232, 2001. Available at: https://doi.org/10.1214/aos/1013203451.

193 Lucie-Smith, L.; Peiris, H. V.; Pontzen, A. An interpretable machine-learning framework for dark matter halo formation. **MNRAS**, v. 490, n. 1, p. 331–342, nov. 2019.

194 GOLOB, A. *et al.* Classifying stars, galaxies, and agns in clauds + hsc-ssp using gradient boosted decision trees. **Monthly Notices of the Royal Astronomical Society**, Oxford University Press (OUP), v. 503, n. 3, p. 4136–4146, Mar 2021. ISSN 1365-2966. Available at: http://dx.doi.org/10.1093/mnras/stab719.

195 Li, N. *et al.* Estimating Dust Attenuation From Galactic Spectra. II. Stellar and Gas Attenuation in Star-forming and Diffuse Ionized Gas Regions in MaNGA. **ApJ**, v. 917, n. 2, p. 72, ago. 2021.

196 Carvajal, R. *et al.* Exploring New Redshift Indicators for Radio-Powerful AGN. **Galaxies**, v. 9, n. 4, p. 86, out. 2021.

197 Cranmer, M. *et al.* Discovering Symbolic Models from Deep Learning with Inductive Biases. **arXiv e-prints**, p. arXiv:2006.11287, jun. 2020.

198 Cranmer, M. Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl. **arXiv e-prints**, p. arXiv:2305.01582, maio 2023.

199 CRANMER, M. **PySR: Fast & Parallelized Symbolic Regression in Python/Julia**. Zenodo, 2020. Available at: http://doi.org/10.5281/zenodo.4041459.

200 CHOLLET, F. *et al.* **Keras**. [*S.l.: s.n.*]: GitHub, 2015. https://github.com/fchollet/keras.

201 ABADI, M. *et al.* **TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems**. 2015. Software available from tensorflow.org. Available at: https://www.tensorflow.org/.

202 PASZKE, A. *et al.* Pytorch: An imperative style, high-performance deep learning library. *In*: **Advances in Neural Information Processing Systems 32**. Curran Associates, Inc., 2019. p. 8024–8035. Available at: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

203 Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization. **arXiv e-prints**, p. arXiv:1412.6980, dez. 2014.

204 Ruder, S. An overview of gradient descent optimization algorithms. **arXiv e-prints**, p. arXiv:1609.04747, set. 2016.

205 LECUN, Y. *et al.* Backpropagation applied to handwritten zip code recognition. **Neural Comput.**, MIT Press, Cambridge, MA, USA, v. 1, n. 4, p. 541–551, dec 1989. ISSN 0899-7667. Available at: https://doi.org/10.1162/neco.1989.1.4.541.

206 DENG, L. The mnist database of handwritten digit images for machine learning research. **IEEE Signal Processing Magazine**, IEEE, v. 29, n. 6, p. 141–142, 2012.

207 Gómez, C. *et al.* Classifying image sequences of astronomical transients with deep neural networks. **MNRAS**, v. 499, n. 3, p. 3130–3138, dez. 2020.

208 Dieleman, S.; Willett, K. W.; Dambre, J. Rotation-invariant convolutional neural networks for galaxy morphology prediction. **MNRAS**, v. 450, n. 2, p. 1441–1459, jun. 2015.

209 Villaescusa-Navarro, F. *et al.* The CAMELS Multifield Data Set: Learning the Universe's Fundamental Parameters with Artificial Intelligence. **ApJS**, v. 259, n. 2, p. 61, abr. 2022.

210 MILANFAR, P. A tour of modern image filtering: New insights and methods, both practical and theoretical. **IEEE Signal Processing Magazine**, v. 30, n. 1, p. 106–128, 2013.

211 FAN, L. *et al.* Brief review of image denoising techniques. **Visual Computing for Industry, Biomedicine, and Art**, v. 2, 2019.

212 Tian, C. *et al.* Deep Learning on Image Denoising: An overview. **arXiv e-prints**, p. arXiv:1912.13171, dez. 2019.

213 Mao, X.-J.; Shen, C.; Yang, Y.-B. Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections. **arXiv e-prints**, p. arXiv:1603.09056, mar. 2016.

214 VINCENT, P. *et al.* Extracting and composing robust features with denoising autoencoders. *In*: . [*S.l.: s.n.*], 2008. p. 1096–1103.

215 Vojtekova, A. *et al.* Learning to denoise astronomical images with U-nets. **MNRAS**, v. 503, n. 3, p. 3204–3215, maio 2021.

216 ZHANG, S. *et al.* Graph convolutional networks: a comprehensive review. **Computational Social Networks**, v. 6, 2019. Available at: https://api.semanticscholar.org/CorpusID:207960027.

217 Zhou, J. *et al.* Graph Neural Networks: A Review of Methods and Applications. **arXiv e-prints**, p. arXiv:1812.08434, dez. 2018.

218 Wu, Z. *et al.* A Comprehensive Survey on Graph Neural Networks. **arXiv e-prints**, p. arXiv:1901.00596, jan. 2019.

219 Bronstein, M. M. *et al.* Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges. **arXiv e-prints**, p. arXiv:2104.13478, abr. 2021.

220 Battaglia, P. W. *et al.* Relational inductive biases, deep learning, and graph networks. **arXiv e-prints**, p. arXiv:1806.01261, jun. 2018.

221 Gilmer, J. *et al.* Neural Message Passing for Quantum Chemistry. **arXiv e-prints**, p. arXiv:1704.01212, abr. 2017.

222 Fey, M.; Lenssen, J. E. Fast Graph Representation Learning with PyTorch Geometric. **arXiv e-prints**, p. arXiv:1903.02428, mar. 2019.

223 Corso, G. *et al.* Principal Neighbourhood Aggregation for Graph Nets. **arXiv e-prints**, p. arXiv:2004.05718, abr. 2020.

224 Wang, Y. *et al.* Dynamic Graph CNN for Learning on Point Clouds. **arXiv e-prints**, p. arXiv:1801.07829, jan. 2018.

225 Kipf, T. N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. **arXiv e-prints**, p. arXiv:1609.02907, set. 2016.

226 Hamilton, W. L.; Ying, R.; Leskovec, J. Inductive Representation Learning on Large Graphs. **arXiv e-prints**, p. arXiv:1706.02216, jun. 2017.

227 Zaheer, M. *et al.* Deep Sets. **arXiv e-prints**, p. arXiv:1703.06114, mar. 2017.

228 Anagnostidis, S. *et al.* Cosmology from Galaxy Redshift Surveys with PointNet. **arXiv e-prints**, p. arXiv:2211.12346, nov. 2022.

229 Massara, E.; Villaescusa-Navarro, F.; Percival, W. J. Predicting Interloper Fraction with Graph Neural Networks. **arXiv e-prints**, p. arXiv:2309.05850, set. 2023.

230 Arbel, J. *et al.* A Primer on Bayesian Neural Networks: Review and Debates. **arXiv e-prints**, p. arXiv:2309.16314, set. 2023.

231 Kobyzev, I.; Prince, S. J. D.; Brubaker, M. A. Normalizing Flows: An Introduction and Review of Current Methods. **arXiv e-prints**, p. arXiv:1908.09257, ago. 2019.

232 Jeffrey, N.; Wandelt, B. D. Solving high-dimensional parameter inference: marginal posterior densities & Moment Networks. **arXiv e-prints**, p. arXiv:2011.05991, nov. 2020.

233 Sadeh, I.; Abdalla, F. B.; Lahav, O. ANNz2: Photometric Redshift and Probability Distribution Function Estimation using Machine Learning. **PASP**, v. 128, n. 968, p. 104502, out. 2016.

234 Pasquet, J. *et al.* Photometric redshifts from SDSS images using a convolutional neural network. **AAP**, v. 621, p. A26, jan. 2019.

235 Lima, E. V. R. *et al.* Photometric redshifts for the S-PLUS Survey: Is machine learning up to the task? **Astronomy and Computing**, v. 38, p. 100510, jan. 2022.

236 KRAWCZYK, B. Learning from imbalanced data: open challenges and future directions. **Progress in Artificial Intelligence**, v. 5, Nov 2016. Available at: https://doi.org/10.1007/s13748-016-0094-0.

237 CHAWLA, N. V. *et al.* Smote: Synthetic minority over-sampling technique. **Journal of Artificial Intelligence Research**, AI Access Foundation, v. 16, p. 321–357, Jun 2002. ISSN 1076-9757. Available at: http://dx.doi.org/10.1613/jair.953.

238 BRANCO, P.; TORGO, L.; RIBEIRO, R. P. SMOGN: a pre-processing approach for imbalanced regression. *In*: TORGO, L. *et al.* (ed.). **Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications**. ECML-PKDD, Skopje, Macedonia: PMLR, 2017. (Proceedings of Machine Learning Research, v. 74), p. 36–50. Available at: http://proceedings.mlr.press/v74/branco17a.html.

239 KUNZ, N. **SMOGN**. [*S.l.: s.n.*]: GitHub, 2019. https://github.com/nickkunz/smogn.

240 BREIMAN, L. Stacked regressions. **Machine Learning**, v. 24, Jul 1996. Available at: https://doi.org/10.1007/BF00117832.

241 GYAMERAH, S. A.; NGARE, P.; IKPE, D. On stock market movement prediction via stacking ensemble learning method. *In*: **2019 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)**. [*S.l.: s.n.*], 2019. p. 1–8.

242 Claesen, M.; De Moor, B. Hyperparameter Search in Machine Learning. **arXiv e-prints**, p. arXiv:1502.02127, fev. 2015.

243 Akiba, T. *et al.* Optuna: A Next-generation Hyperparameter Optimization Framework. **arXiv e-prints**, p. arXiv:1907.10902, jul. 2019.

244 BERGSTRA, J. *et al.* Algorithms for hyper-parameter optimization. *In*: SHAWE-TAYLOR, J. *et al.* (ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2011. v. 24. Available at: https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf.

245 Watanabe, S. Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance. **arXiv e-prints**, p. arXiv:2304.11127, abr. 2023.

246 SANTI, N. S. M. de; ABRAMO, L. R. Primeiros passos na obtenc̃ão de parâmetros cosmológicos utilizando matrizes de covariância cosmológicas sem ruído. *In*: **Anais do I Encontro Brasileiro de Meninas e Mulheres da Astrofísica, Gravitacão e Cosmologia - As Astrocientistas**. Blucher, 2022. p. 93–101. Available at: https://www.proceedings.blucher.com.br/article-details/primeiros-passos-na-obteno-de-parmetros-cosmolgicos-utilizando-matrizes-de-covarincia-cosmolgicas-se

247 SANTI, N. S. M. de; ABRAMO, L. R. Obtaining cosmological covariance matrices with machine learning. *In*: **Boletim da Sociedade Astronômica Brasileira**. Boletim da SAB, 2023. v. 34, n. 1, p. 193–197. Available at: https://sab-astro.org.br/wp-content/uploads/2023/04/NataliSolerMatubarodeSanti.pdf.

248 Hartlap, J.; Simon, P.; Schneider, P. Why your model parameter confidences might be too optimistic. Unbiased estimation of the inverse covariance matrix. **AAP**, v. 464, n. 1, p. 399–404, mar. 2007.

249 Dodelson, S.; Schneider, M. D. The effect of covariance estimator error on cosmological parameter constraints. **PRD**, v. 88, n. 6, p. 063537, set. 2013.

250 Heavens, A. F. *et al.* Massive data compression for parameter-dependent covariance matrices. **MNRAS**, v. 472, n. 4, p. 4244–4250, dez. 2017.

251 Blot, L. *et al.* Non-linear matter power spectrum covariance matrix errors and cosmological parameter uncertainties. **MNRAS**, v. 458, n. 4, p. 4462–4470, jun. 2016.

252 Meiksin, A.; White, M. The growth of correlations in the matter power spectrum. **MNRAS**, v. 308, n. 4, p. 1179–1184, out. 1999.

253 Fumagalli, A. *et al.* Fitting covariance matrix models to simulations. **JCAP**, v. 2022, n. 12, p. 022, dez. 2022.

254 Heavens, A. F.; Jimenez, R.; Lahav, O. Massive lossless data compression and multiple parameter estimation from galaxy spectra. **MNRAS**, v. 317, n. 4, p. 965–972, out. 2000.

255 PHILCOX, O. H. E. *et al.* Fewer mocks and less noise: Reducing the dimensionality of cosmological observables with subspace projections. **Phys. Rev. D**, American Physical Society, v. 103, p. 043508, Feb 2021. Available at: https://link.aps.org/doi/10.1103/PhysRevD.103.043508.

256 Schneider, M. D. *et al.* Fast Generation of Ensembles of Cosmological N-body Simulations Via Mode Resampling. **ApJ**, v. 737, n. 1, p. 11, ago. 2011.

257 Chartier, N. *et al.* CARPool: fast, accurate computation of large-scale structure statistics by pairing costly and cheap cosmological simulations. **MNRAS**, v. 503, n. 2, p. 1897–1914, maio 2021.

258 Chartier, N.; Wandelt, B. D. CARPool covariance: fast, unbiased covariance estimation for large-scale structure observables. **MNRAS**, v. 509, n. 2, p. 2220–2233, jan. 2022.

259 Ni, Y. *et al.* AI-assisted superresolution cosmological simulations - II. Halo substructures, velocities, and higher order statistics. **MNRAS**, v. 507, n. 1, p. 1021–1033, out. 2021.

260 Kodi Ramanah, D. *et al.* Super-resolution emulator of cosmological simulations using deep physical models. **MNRAS**, v. 495, n. 4, p. 4227–4236, jul. 2020.

261 He, S. *et al.* Learning to predict the cosmological structure formation. **Proceedings of the National Academy of Science**, v. 116, n. 28, p. 13825–13832, jul. 2019.

262 Alves de Oliveira, R. *et al.* Fast and Accurate Non-Linear Predictions of Universes with Deep Learning. **arXiv e-prints**, p. arXiv:2012.00240, nov. 2020.

263 Kaushal, N. *et al.* NECOLA: Toward a Universal Field-level Cosmological Emulator. **ApJ**, v. 930, n. 2, p. 115, maio 2022.

264 Davis, M. *et al.* The evolution of large-scale structure in a universe dominated by cold dark matter. **APJ**, v. 292, p. 371–394, maio 1985.

265 Diemer, B. COLOSSUS: A Python Toolkit for Cosmology, Large-scale Structure, and Dark Matter Halos. **APJS**, v. 239, n. 2, p. 35, dez. 2018.

266 Coupon, J. *et al.* The bright-star masks for the HSC-SSP survey. **PASJ**, v. 70, p. S7, jan. 2018.

267 Valcin, D. *et al.* BE-HaPPY: bias emulator for halo power spectrum including massive neutrinos. **JCAP**, v. 2019, n. 12, p. 057, dez. 2019.

268 Vogeley, M. S.; Szalay, A. S. Eigenmode Analysis of Galaxy Redshift Surveys. I. Theory and Methods. **ApJ**, v. 465, p. 34, jul. 1996.

269 Ferreira, T. *et al.* Data compression and covariance matrix inspection: Cosmic shear. **PRD**, v. 103, n. 10, p. 103535, maio 2021.

270 WISHART, J. The generalised product moment distribution in samples from a normal multivariate population. **Biometrika**, [Oxford University Press, Biometrika Trust], v. 20A, n. 1/2, p. 32–52, 1928. ISSN 00063444. Available at: http://www.jstor.org/stable/2331939.

271 Sellentin, E.; Heavens, A. F. Parameter inference with estimated covariance matrices. **MNRAS**, v. 456, n. 1, p. L132–L136, fev. 2016.

272 Hall, A.; Taylor, A. A Bayesian method for combining theoretical and simulated covariance matrices for large-scale structure surveys. **MNRAS**, v. 483, n. 1, p. 189–207, fev. 2019.

273 Foreman-Mackey, D. *et al.* emcee: The MCMC Hammer. **PASP**, v. 125, n. 925, p. 306, mar. 2013.

274 CALDERON, V. F.; BERLIND, A. A. Prediction of galaxy halo masses in sdss dr7 via a machine learning approach. **Monthly Notices of the Royal Astronomical Society**, v. 490, p. 2367–2379, 2019.

275 MAN, Z.-Y. *et al.* The fundamental relation between halo mass and galaxy group properties. **The Astrophysical Journal**, v. 881, 2019.

276 SANTI, N. S. M. d. *et al.* A machine learning suite to halo-galaxy connection. *In*: BUFANO, F. *et al.* (ed.). **Machine Learning for Astrophysics**. Cham: Springer International Publishing, 2023. p. 31–34. ISBN 978-3-031-34167-0.

277 Chuang, C.-Y. *et al.* Leaving No Branches Behind: Predicting Baryonic Properties of Galaxies from Merger Trees. **arXiv e-prints**, p. arXiv:2311.09162, nov. 2023.

278 Pillepich, A. *et al.* Simulating galaxy formation with the IllustrisTNG model. **MNRAS**, v. 473, n. 3, p. 4077–4106, jan. 2018.

279 Pillepich, A. *et al.* First results from the IllustrisTNG simulations: the stellar mass content of groups and clusters of galaxies. **MNRAS**, v. 475, n. 1, p. 648–675, mar. 2018.

280 Nelson, D. *et al.* First results from the IllustrisTNG simulations: the galaxy colour bimodality. **MNRAS**, v. 475, n. 1, p. 624–647, mar. 2018.

281 NELSON, D. *et al.* The illustristng simulations: public data release. **Computational Astrophysics and Cosmology**, v. 6, 2019.

282 Marinacci, F. *et al.* First results from the IllustrisTNG simulations: radio haloes and magnetic fields. **MNRAS**, v. 480, n. 4, p. 5113–5139, nov. 2018.

283 Naiman, J. P. *et al.* First results from the IllustrisTNG simulations: a tale of two elements - chemical evolution of magnesium and europium. **MNRAS**, v. 477, n. 1, p. 1206–1224, jun. 2018.

284 Springel, V. *et al.* First results from the IllustrisTNG simulations: matter and galaxy clustering. **MNRAS**, v. 475, n. 1, p. 676–698, Mar 2018.

285 Vogelsberger, M. *et al.* Introducing the Illustris Project: simulating the coevolution of dark and visible matter in the Universe. **MNRAS**, v. 444, n. 2, p. 1518–1547, Oct 2014.

286 Vogelsberger, M. *et al.* Properties of galaxies reproduced by a hydrodynamic simulation. **NAT**, v. 509, n. 7499, p. 177–182, May 2014.

287 Genel, S. *et al.* Introducing the Illustris project: the evolution of galaxy populations across cosmic time. **MNRAS**, v. 445, n. 1, p. 175–200, Nov 2014.

288 Bose, S. *et al.* Revealing the galaxy-halo connection in IllustrisTNG. **MNRAS**, p. 2192, Sep 2019.

289 Beltz-Mohrmann, G. D.; Berlind, A. A.; Szewciw, A. O. Testing the accuracy of halo occupation distribution modelling using hydrodynamic simulations. **MNRAS**, v. 491, n. 4, p. 5771–5788, fev. 2020.

290 Contreras, S.; Angulo, R.; Zennaro, M. A flexible modelling of galaxy assembly bias. **arXiv e-prints**, p. arXiv:2005.03672, maio 2020.

291 Gu, M. *et al.* Coordinated Assembly of Galaxy Groups and Clusters in the IllustrisTNG Simulations. **arXiv e-prints**, p. arXiv:2010.04166, out. 2020.

292 Hadzhiyska, B. *et al.* Limitations to the 'basic' HOD model and beyond. **MNRAS**, v. 493, n. 4, p. 5506–5519, abr. 2020.

293 Montero-Dorta, A. D. *et al.* The manifestation of secondary bias on the galaxy population from IllustrisTNG300. **MNRAS**, v. 496, n. 2, p. 1182–1196, ago. 2020.

294 Bullock, J. S. *et al.* A Universal Angular Momentum Profile for Galactic Halos. **ApJ**, v. 555, n. 1, p. 240–257, jul. 2001.

295 Navarro, J. F.; Frenk, C. S.; White, S. D. M. A Universal Density Profile from Hierarchical Clustering. **APJ**, v. 490, n. 2, p. 493–508, dez. 1997.

296 Artale, M. C. *et al.* The impact of assembly bias on the halo occupation in hydrodynamical simulations. **MNRAS**, v. 480, n. 3, p. 3978–3992, nov. 2018.

297 Buser, R. A systematic investigation of multicolor photometric systems. I. The UBV, RGU and uvby systems. **AAP**, v. 62, p. 411–424, jan. 1978.

298 Favole, G. *et al.* SHAM through the lens of a hydrodynamical simulation. **arXiv e-prints**, p. arXiv:2101.10733, jan. 2021.

299 Peacock, J. A. Two-dimensional goodness-of-fit testing in astronomy. **MNRAS**, v. 202, p. 615–627, fev. 1983.

300 Fasano, G.; Franceschini, A. A multidimensional version of the Kolmogorov-Smirnov test. **MNRAS**, v. 225, p. 155–170, mar. 1987.

301 TAILLON, G. **2DKS**. [*S.l.: s.n.*]: GitHub, 2018. https://github.com/Gabinou/2DKS.

302 Rodriguez, F. *et al.* The galaxy size-halo mass scaling relations and clustering properties of central and satellite galaxies. **MNRAS**, v. 505, n. 3, p. 3192–3205, ago. 2021.

303 Montero-Dorta, A. D. *et al.* On the influence of halo mass accretion history on galaxy properties and assembly bias. **MNRAS**, v. 508, n. 1, p. 940–949, nov. 2021.

304 Peebles, P. J. E. **The large-scale structure of the universe**. [*S.l.: s.n.*]: Princeton University Press, 1980.

305 Kaiser, N. Clustering in real space and in redshift space. **MNRAS**, v. 227, p. 1–21, jul. 1987.

306 Sargent, W. L. W.; Turner, E. L. A statistical method for determining the cosmological density parameter from the redshifts of a complete sample of galaxies. **APJL**, v. 212, p. L3–L7, fev. 1977.

307 TONEGAWA, M. *et al.* Cosmological information from the small-scale redshift-space distortion. **The Astrophysical Journal**, American Astronomical Society, v. 897, n. 1, p. 17, jun 2020. Available at: https://doi.org/10.3847%2F1538-4357%2Fab95ff.

308 Cen, R.; Bahcall, N. A.; Gramann, M. Velocity Correlations of Galaxy Clusters. **APJL**, v. 437, p. L51, dez. 1994.

309 Ma, Y.-Z.; Li, M.; He, P. Constraining cosmology with pairwise velocity estimator. **AAP**, v. 583, p. A52, nov. 2015.

310 Shao, H. *et al.* A Universal Equation to Predict Ω_m from Halo and Galaxy Catalogs. **ApJ**, v. 956, n. 2, p. 149, out. 2023.

311 Ni, Y. *et al.* The CAMELS Project: Expanding the Galaxy Formation Model Space with New ASTRID and 28-parameter TNG and SIMBA Suites. **ApJ**, v. 959, n. 2, p. 136, dez. 2023.

312 Wu, J. F.; Kragh Jespersen, C.; Wechsler, R. H. How the Galaxy-Halo Connection Depends on Large-Scale Environment. **arXiv e-prints**, p. arXiv:2402.07995, fev. 2024.

313 Lemos, P. *et al.* SimBIG: Field-level Simulation-based Inference of Large-scale Structure. *In*: **Machine Learning for Astrophysics**. [*S.l.: s.n.*], 2023. p. 18.

314 Villaescusa-Navarro, F. *et al.* The CAMELS Project: Cosmology and Astrophysics with Machine-learning Simulations. **ApJ**, v. 915, n. 1, p. 71, jul. 2021.

315 Villaescusa-Navarro, F. *et al.* The CAMELS project: public data release. **arXiv e-prints**, p. arXiv:2201.01300, jan. 2022.

316 Weinberger, R. *et al.* Simulating galaxy formation with black hole driven thermal and kinetic feedback. **MNRAS**, v. 465, p. 3291–3308, mar. 2017.

317 Schaller, M. *et al.* SWIFT: Using Task-Based Parallelism, Fully Asynchronous Communication, and Graph Partition-Based Domain Decomposition for Strong Scaling on more than 100,000 Cores. *In*: **Proceedings of the Platform for Advanced Scientific Computing Conference**. [*S.l.: s.n.*], 2016. p. 2.

318 SOBOL', I. On the distribution of points in a cube and the approximate evaluation of integrals. **USSR Computational Mathematics and Mathematical Physics**, v. 7, n. 4, p. 86–112, 1967. ISSN 0041-5553. Available at: https://www.sciencedirect.com/science/article/pii/0041555367901449.

319 Springel, V. E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh. **MNRAS**, v. 401, n. 2, p. 791–851, jan. 2010.

320 Weinberger, R.; Springel, V.; Pakmor, R. The AREPO Public Code Release. **APJS**, v. 248, n. 2, p. 32, jun. 2020.

321 HOPKINS, P. F. A new class of accurate, mesh-free hydrodynamic simulation methods. **Monthly Notices of the Royal Astronomical Society**, v. 450, n. 1, p. 53–110, 04 2015. ISSN 0035-8711. Available at: https://doi.org/10.1093/mnras/stv195.

322 Feng, Y. *et al.* **MP-Gadget/MP-Gadget: A tag for getting a DOI**. Zenodo, 2018. Available at: https://doi.org/10.5281/zenodo.1451799.

323 Ni, Y. *et al.* The ASTRID simulation: the evolution of supermassive black holes. **MNRAS**, v. 513, n. 1, p. 670–692, jun. 2022.

324 Springel, V. The cosmological simulation code GADGET-2. **MNRAS**, v. 364, n. 4, p. 1105–1134, dez. 2005.

325 Springel, V.; Hernquist, L. Cosmological smoothed particle hydrodynamics simulations: the entropy equation. **MNRAS**, v. 333, n. 3, p. 649–664, jul. 2002.

326 Dolag, K. *et al.* Thermal Conduction in Simulated Galaxy Clusters. **APJL**, v. 606, n. 2, p. L97–L100, maio 2004.

327 Dolag, K. *et al.* Turbulent gas motions in galaxy cluster simulations: the role of smoothed particle hydrodynamics viscosity. **MNRAS**, v. 364, n. 3, p. 753–772, dez. 2005.

328 Dolag, K. *et al.* Simulating the physical properties of dark matter and gas inside the cosmic web. **MNRAS**, v. 370, n. 2, p. 656–672, ago. 2006.

329 Springel, V.; Hernquist, L. Cosmological smoothed particle hydrodynamics simulations: a hybrid multiphase model for star formation. **MNRAS**, v. 339, n. 2, p. 289–311, fev. 2003.

330 Springel, V.; Di Matteo, T.; Hernquist, L. Modelling feedback from stars and black holes in galaxy mergers. **MNRAS**, v. 361, n. 3, p. 776–794, ago. 2005.

331 Di Matteo, T.; Springel, V.; Hernquist, L. Energy input from quasars regulates the growth and activity of black holes and their host galaxies. **NAT**, v. 433, n. 7026, p. 604–607, fev. 2005.

332 Fabjan, D. *et al.* X-ray mass proxies from hydrodynamic simulations of galaxy clusters - I. **MNRAS**, v. 416, n. 2, p. 801–816, set. 2011.

333 Hirschmann, M. *et al.* Cosmological simulations of black hole growth: AGN luminosities and downsizing. **MNRAS**, v. 442, n. 3, p. 2304–2324, ago. 2014.

334 Steinborn, L. K. *et al.* Origin and properties of dual and offset active galactic nuclei in a cosmological simulation at z=2. **MNRAS**, v. 458, n. 1, p. 1013–1028, maio 2016.

335 Schaller, M. *et al.* **SWIFT: SPH With Inter-dependent Fine-grained Tasking**. 2018. ascl:1805.020 p. Astrophysics Source Code Library.

336 Crain, R. A. *et al.* The EAGLE simulations of galaxy formation: calibration of subgrid physics and model variations. **MNRAS**, v. 450, n. 2, p. 1937–1961, jun. 2015.

337 Borrow, J. *et al.* The impact of stochastic modeling on the predictive power of galaxy formation simulations. **arXiv e-prints**, p. arXiv:2211.08442, nov. 2022.

338 Weinberger, R. *et al.* Simulating galaxy formation with black hole driven thermal and kinetic feedback. **MNRAS**, v. 465, n. 3, p. 3291–3308, mar. 2017.

339 Nelson, D. *et al.* First results from the IllustrisTNG simulations: the galaxy colour bimodality. **MNRAS**, v. 475, n. 1, p. 624–647, mar. 2018.

340 Li, H. *et al.* Visualizing the Loss Landscape of Neural Nets. **arXiv e-prints**, p. arXiv:1712.09913, dez. 2017.

341 Hamilton, A. J. S.; Rimes, C. D.; Scoccimarro, R. On measuring the covariance matrix of the non-linear power spectrum from simulations. **MNRAS**, v. 371, n. 3, p. 1188–1204, set. 2006.

342 Takada, M.; Bridle, S. Probing dark energy with cluster counts and cosmic shear power spectra: including the full covariance. **New Journal of Physics**, v. 9, n. 12, p. 446, dez. 2007.

343 Li, Y.; Hu, W.; Takada, M. Super-sample covariance in simulations. **PRD**, v. 89, n. 8, p. 083519, abr. 2014.

344 Abramo, L. R.; Secco, L. F.; Loureiro, A. Fourier analysis of multitracer cosmological surveys. **MNRAS**, v. 455, n. 4, p. 3871–3889, fev. 2016.

345 Hassani, H.; Javanmard, A. The curse of overparametrization in adversarial training: Precise analysis of robust generalization for random features regression. **arXiv e-prints**, p. arXiv:2201.05149, jan. 2022.

346 Howlett, C. *et al.* The Sloan Digital Sky Survey peculiar velocity catalogue. **MNRAS**, v. 515, n. 1, p. 953–976, set. 2022.

347 Kourkchi, E. *et al.* Cosmicflows-4: The Catalog of ∼10,000 Tully-Fisher Distances. **ApJ**, v. 902, n. 2, p. 145, out. 2020.

348 Garrison, L. H. *et al.* The ABACUS cosmological N-body code. **MNRAS**, v. 508, n. 1, p. 575–596, nov. 2021.

349 Harnois-Déraps, J. *et al.* High-performance P^3M N-body code: CUBEP^{3}M. **MNRAS**, v. 436, n. 1, p. 540–559, nov. 2013.

350 Bryan, G. L. *et al.* ENZO: An Adaptive Mesh Refinement Code for Astrophysics. **APJS**, v. 211, n. 2, p. 19, abr. 2014.

351 Hockney, R. W.; Eastwood, J. W. **Computer simulation using particles**. [*S.l.: s.n.*], 1988.

352 Greengard, L.; Rokhlin, V. A Fast Algorithm for Particle Simulations. **Journal of Computational Physics**, v. 73, n. 2, p. 325–348, dez. 1987.

353 Potter, D.; Stadel, J.; Teyssier, R. PKDGRAV3: beyond trillion particle cosmological simulations for the next era of galaxy surveys. **Computational Astrophysics and Cosmology**, v. 4, n. 1, p. 2, maio 2017.

354 Teyssier, R. Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES. **AAP**, v. 385, p. 337–364, abr. 2002.

355 Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. **arXiv e-prints**, p. arXiv:1711.05101, nov. 2017.

356 BERNARDEAU, F. *et al.* Omega from the skewness of the cosmic velocity divergence. **Monthly Notices of the Royal Astronomical Society**, Oxford University Press (OUP), v. 274, n. 1, p. 20–26, may 1995. Available at: https://doi.org/10.1093%2Fmnras%2F274.1.20.

357 Dekel, A. Dynamics of Cosmic Flows. **ARAA**, v. 32, p. 371–418, jan. 1994.

358 JUSZKIEWICZ, R.; SPRINGEL, V.; DURRER, R. Dynamics of pairwise motions. **The Astrophysical Journal**, American Astronomical Society, v. 518, n. 1, p. L25–L28, jun 1999. Available at: https://doi.org/10.1086%2F312055.

359 JUSZKIEWICZ, R. *et al.* Evidence for a low-density universe from the relative velocities of galaxies. **Science**, American Association for the Advancement of Science (AAAS), v. 287, n. 5450, p. 109–112, jan 2000. Available at: https://doi.org/10.1126%2Fscience.287.5450.109.

360 Heymans, C. *et al.* CFHTLenS: the Canada-France-Hawaii Telescope Lensing Survey. **MNRAS**, v. 427, n. 1, p. 146–166, nov. 2012.

361 Coupon, J. *et al.* Photometric redshifts for the CFHTLS T0004 deep and wide fields. **AAP**, v. 500, n. 3, p. 981–998, jun. 2009.

362 Howlett, C.; Staveley-Smith, L.; Blake, C. Cosmological forecasts for combined and next-generation peculiar velocity surveys. **MNRAS**, v. 464, n. 3, p. 2517–2544, jan. 2017.

363 Tonry, J.; Schneider, D. P. A New Technique for Measuring Extragalactic Distances. **AJ**, v. 96, p. 807, set. 1988.

364 Tully, R. B.; Fisher, J. R. A new method of determining distances to galaxies. **AAP**, v. 54, p. 661–673, fev. 1977.

365 Bahcall, N. A.; Oh, S. P. The Peculiar Velocity Function of Galaxy Clusters. **ApJL**, v. 462, p. L49, maio 1996.

366 Lan, T.-W. *et al.* The DESI Survey Validation: Results from Visual Inspection of Bright Galaxies, Luminous Red Galaxies, and Emission-line Galaxies. **ApJ**, v. 943, n. 1, p. 68, jan. 2023.

367 Nelson, D. *et al.* First results from the IllustrisTNG simulations: the galaxy colour bimodality. **MNRAS**, v. 475, n. 1, p. 624–647, mar. 2018.

368 Donnari, M. *et al.* The star formation activity of IllustrisTNG galaxies: main sequence, UVJ diagram, quenched fractions, and systematics. **MNRAS**, v. 485, n. 4, p. 4817–4840, jun. 2019.

369 Andersen, P.; Davis, T. M.; Howlett, C. Cosmology with peculiar velocities: observational effects. **MNRAS**, v. 463, n. 4, p. 4083–4092, dez. 2016.

370 Ćiprijanović, A. *et al.* DeepAstroUDA: semi-supervised universal domain adaptation for cross-survey galaxy morphology classification and anomaly detection. **Machine Learning: Science and Technology**, v. 4, n. 2, p. 025013, jun. 2023.

371 Roncoli, A. *et al.* Domain Adaptive Graph Neural Networks for Constraining Cosmological Parameters Across Multiple Data Sets. **arXiv e-prints**, p. arXiv:2311.01588, nov. 2023.

372 Echeverri-Rojas, N. *et al.* Cosmology with One Galaxy? The ASTRID Model and Robustness. **ApJ**, v. 954, n. 2, p. 125, set. 2023.

APPENDIX

APPENDIX A – SOLVING THE POISSON EQUATION IN FOURIER SPACE

This appendix is related to Section 2.5.1 and it shows the details behind the solution of Equation 2.76. First, we can write the *discretized version* of Poisson's equation, using the 7-point "crest" template (127, 128), as

$$\tilde{\nabla}^2\tilde{\phi} \simeq \tilde{\phi}_{i-1,j,k} + \tilde{\phi}_{i+1,j,k} + \tilde{\phi}_{i,j-1,k} + \tilde{\phi}_{i,j+1,k} + \tilde{\phi}_{i,j,k-1} + \tilde{\phi}_{i,j,k+1} - 6\tilde{\phi}_{i,j,k} = \frac{3}{2}\frac{\Omega_{m,0}}{a}\tilde{\delta}, \quad \text{(A.1)}$$

where $(i, j, k) = 1, \ldots, N_g$. Then, we write the fields $\tilde{\phi}(\mathbf{k})$, $\tilde{\phi}(\mathbf{x})$, and $\tilde{\delta}(\mathbf{x})$ in the discrete space according to

$$\tilde{\phi}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z} = \frac{1}{N_g^{3/2}} \sum_{i,j,k=0}^{N_g-1} \exp\left[i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y j + \bar{k}_z k\right)\right] \tilde{\phi}^{\mathbf{x}}_{i,j,k} \quad \text{(A.2)}$$

$$\tilde{\phi}^{\mathbf{x}}_{i,j,k} = \frac{1}{N_g^{3/2}} \sum_{\bar{k}_x,\bar{k}_y,\bar{k}_z=0}^{N_g-1} \exp\left[-i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y j + \bar{k}_z k\right)\right] \tilde{\phi}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z} \quad \text{(A.3)}$$

$$\tilde{\delta}^{\mathbf{x}}_{i,j,k} = \frac{1}{N_g^{3/2}} \sum_{\bar{k}_x,\bar{k}_y,\bar{k}_z=0}^{N_g-1} \exp\left[-i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y j + \bar{k}_z k\right)\right] \tilde{\delta}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z}, \quad \text{(A.4)}$$

where $\tilde{\phi}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z} \leftrightarrow \tilde{\phi}(\mathbf{k})$, $\tilde{\phi}^{\mathbf{x}}_{i,j,k} \leftrightarrow \tilde{\phi}(\mathbf{x})$, $\tilde{\delta}^{\mathbf{x}}_{i,j,k} \leftrightarrow \tilde{\delta}(\mathbf{x})$, and N_g means we are using code units in the way that $L_{BOX} = N_g$.

Replacing the above fields in the Equation A.1 we have

$$\begin{aligned}\frac{1}{N_g^{3/2}} \sum_{\bar{k}_x,\bar{k}_y,\bar{k}_z=0}^{N_g-1} \Bigg\{ &\exp\left[-i\frac{2\pi}{N_g}\left(\bar{k}_x(i \pm 1) + \bar{k}_y j + \bar{k}_z k\right)\right] + \exp\left[-i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y(j \pm 1) + \bar{k}_z k\right)\right] \\ &+ \exp\left[-i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y j + \bar{k}_z(k \pm 1)\right)\right] - 6\exp\left[-i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y j + \bar{k}_z k\right)\right] \Bigg\} \tilde{\phi}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z} = \\ &= \frac{3}{2}\frac{\Omega_{m,0}}{a}\frac{1}{N_g^{3/2}} \sum_{\bar{k}_x,\bar{k}_y,\bar{k}_z=0}^{N_g-1} \exp\left[-i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y j + \bar{k}_z k\right)\right] \tilde{\delta}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z}. \quad \text{(A.5)}\end{aligned}$$

Multiplying both sides of Equation A.5 by $\exp\left[i\frac{2\pi}{N_g}\left(\bar{k}_x i + \bar{k}_y j \ \ + \bar{k}_z k\right)\right]$, we get

$$\begin{aligned}\left[\left(e^{i\frac{2\pi}{N_g}\bar{k}_x} + e^{-i\frac{2\pi}{N_g}\bar{k}_x}\right) + \left(e^{i\frac{2\pi}{N_g}\bar{k}_y} + e^{-i\frac{2\pi}{N_g}\bar{k}_y}\right) + \left(e^{i\frac{2\pi}{N_g}\bar{k}_z} + e^{-i\frac{2\pi}{N_g}\bar{k}_z}\right) - 6\right] \tilde{\phi}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z} = \\ = \frac{3}{2}\frac{\Omega_{m,0}}{a}\tilde{\delta}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z}. \quad \text{(A.6)}\end{aligned}$$

Converting the exponential terms as

$$e^{i\frac{2\pi}{N_g}\bar{k}_x} + e^{-i\frac{2\pi}{N_g}\bar{k}_x} - 2 = 2\left[\cos\left(\frac{2\pi}{N_g}\bar{k}_x\right) - 1\right] = -4\sin^2\left(\frac{\pi\bar{k}_x}{N_g}\right), \quad \text{(A.7)}$$

we have

$$-4\left[\sin^2\left(\frac{\pi\bar{k}_x}{N_g}\right)+\sin^2\left(\frac{\pi\bar{k}_y}{N_g}\right)+\sin^2\left(\frac{\pi\bar{k}_z}{N_g}\right)\right]\tilde{\phi}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z}=\frac{3}{2}\frac{\Omega_{m,0}}{a}\tilde{\delta}^{\mathbf{k}}_{\bar{k}_x,\bar{k}_y,\bar{k}_z}. \tag{A.8}$$

And, finally

$$\tilde{\phi}(\mathbf{k})=G(\mathbf{k})\tilde{\delta}(\mathbf{k}), \tag{A.9}$$

with

$$G(\mathbf{k})=-\frac{3\Omega_{m,0}}{8a}\left[\sin^2\left(\frac{k_x}{2}\right)+\sin^2\left(\frac{k_y}{2}\right)+\sin^2\left(\frac{k_z}{2}\right)\right]^{-1}, \tag{A.10}$$

where $k_{x,y,z}=2\pi\bar{k}_{x,y,z}/N_g$, exactly as presented in Equation (2.85).